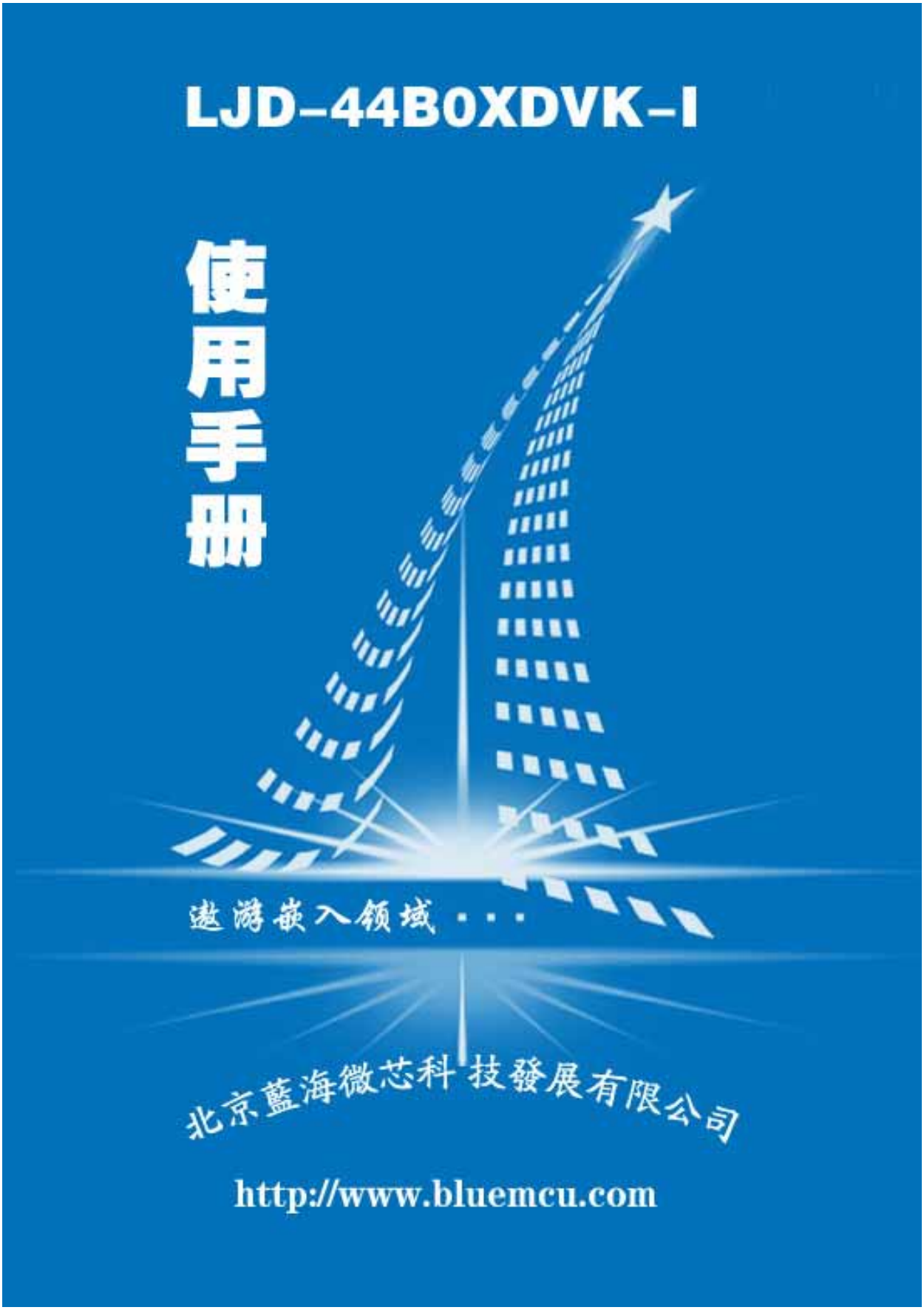
LJD-44B0XDVK-I

使用手册



目　　录

第一章：LJD-44B0XDVK-I 开发套件组成部分 4

1．1 学习板相关配套硬件清单 4
1．2　学习板相关配套软件件清单 4

第二章：LJD-44B0XDVK-I 硬件资源描述 5

2.1 开发板硬件资源及接口分配 5
2.2 板上指示灯功能及跳线功能说明 5
2.3 开发板接口使用说明 5
2.4 开发板硬件资源说明 6
2.5 学习板核心板硬件电路说明 7
2.5.1 CPU 单元复位电路及电源电路 7
2.5.2 NORFLASH 和 NANDFLASH 电路 7
2.5.3 网卡接口芯片电路 8
2.5.4 SDRAM 电路 8

第三章．开发板各个模块电路实验测试方法 9

3.1．如何运行测试程序 9
3.3 内存 SDRAM 读写自测试 11
3.4 PWM 脉宽调试和蜂鸣器测试 11
3.5 IIC 总线 EEPROM 读写测试 12
3.6 模数转换器 ADC 测试 13
3.7 IIS 音频播放 WAV 文件测试 14
3.8 USB DEVICE（PDIUSBD12）测试 16
3.9 黑白 STN 液晶屏测试 17
3.10 黑白 STN 液晶屏显示英文字符测试 17
3.11 4 级灰度 STN 液晶屏测试 18
3.12 16 级灰度 STN 液晶屏测试 19
3.13 256 色 STN 液晶屏测试 20
3.14 外部电平中断测试 20

第四章．ADS 开发环境使用介绍 21

4.1 使用 ADS 创建工程 21
4.1.1 建立一个工程 21
4.1.2 编译和连接工程 24
4.2 用 AXD 进行代码调试 29

4.2.1 安装并运行 ARM-JTAG 调试代理 ARM7 请看多媒体演示程序 29
4.2.2 为 ARM-JTAG 调试代理正确配置 AXD DEBUGGER 请看多媒体演示程序 29
4.2.3 使用 SUPERJTAG 在 ADS1.2 环境下进行仿真调试 关闭并重新启动 AXD DEBUGGER，点击菜单 FILE/LOAD IMAGE，找到您想调试的调试目标代码（*.AXF 格式），打开它就启动目标代码下载，这时会出现进度条，下载完毕就可以单步或者全速调试了，调试过程中可以看 CPU 各寄存器，也可以设置断点，可以单步，全速等等调试功能。 29

第五章．学习板相关程序烧写及 BIOS 使用方法 30

5．1 BIOS 烧写方法 30
5．2 BIOS 备份 30
5．3 用 BIOS 通过网口烧写 UC/OS-II 应用程序到 FLASH 31
5．4 用 BIOS 通过网口快速运行和调试 UC/OS-II 应用程序 33
5．5 BIOS 相关命详解 33

第六章．UCLINUX 开发环境的建立和使用方法.....38

6．1 UCLINUX 开发环境如何建立 38
6．2 UCLINUX 内核编译步骤 38
6．3 UCLINUX 如何烧写 43
6.3.1 烧写中断向量表 BOOT.BIN 43
6.3.2 烧写 UCLINUX 内核 43
6.3.3 烧写 UCLINUX 根文件系统 44
6.3.4 输入命令 MRUN 运行 UCLINUX 45
6．4 测试目标 45

第七章．学习板常见故障及解决方法.....47

7.1 如何重新烧写 BIOS? 47
7.2 为何超级终端不能输入？ 47
7.3 为何超级终端里，BIOS 不接受 AP 等指令？ 47
7.4 为何网络 PING 不通？ 47
7.5 为何 SUPERJTAG 连不上目标板？ 47
7.6 FLASHPGM 无法进行 FLASH 烧写怎么办？ 48

第一章：LJD-44b0XDVK-I 开发套件组成部分

1．1 学习板相关配套硬件清单

1).LJD-44B0XDVK-I 开发板一块(由核心板 4 层工艺和底板组成)
2).LJD-44B0XDVK-I 光盘一张
3).JTAG 调试头一个
4).一条并口线（一边是公头一边是母头，一对一）
5).一条串口线(两边都是母头，交叉串口线)
6).一条网线(交叉网线线)
7).5V 稳压电源一个

1．2　学习板相关配套软件件清单

1).ADS1.2 安装软件（评估版）
2).使用SUPERJTAG 并支持ADS1.20 和SDT2.51 的JTAG 调试软件ARMJTAGDEBUGFINAL；
3).烧写FLASH 的工具软件FLASHPGM2.2.4（评估版）以及适用于S3C44B0X 的OCD 配置文件；
4).串口工具软件sscom32.exe、dnw.exe、tftp.exe；
5).LCD 图片转换和字模转换工具软件UC-GUI-BITMAPCONVERT.EXE；
6).LJD-44B0XDVK-I BIOS 源代码（ADS1.20 的项目文件，包含CS8900 驱动和TFTP 协议源码）；
7).各个模块实验综合程序；
8).UC/OS-II　测试（ADS1.2 的项目文件，三个任务的演示DEMO，提供全部源码）
9).UCLINUX 源码包和编译器等；
10).使用SUPERJTAG 进行硬件仿真DEBUG 的演示动画；
11).使用FLASHPGM 烧写FLASH 的演示动画；
12).LJD-44B0XDVK-I 原理图（pdf 格式）；
13).LJD-44B0XDVK-I 学习板使用手册（pdf 格式）

第二章：LJD-44b0XDVK-I 硬件资源描述

2.1 开发板硬件资源及接口分配

1).系统片选及地址分配

地址区间	说明	片选信号
0x00000000~0x02000000	FLASH 存储器：sst39vf1601	nGCS1
0x02000000-0x0200004	NAND FLASH	nGCS1
0x0400 0020~0x0400 002e之间的偶数地址	----------	nGCS2
0x0600 0000~0x0600 0001	网络接口芯片 CS8900 的读/写	nGCS3
0x0800 0000~0x0800 0001	USB 接口芯片 PDIUSBD12 读/写	nGCS4
0x0A000000~0x0C00 0000	----------	nGCS5
0x0C000000~0x0C7FFFFF	存储器空间：8M Byte	nGCS6
0x0e000000-	----------	nGCS7

2). A/D 采样接口

A/D 口	说明
AIN0-AIN7	外接输入

2.2 板上指示灯功能及跳线功能说明

1).核心板发光二极管 D3 2.5V 电源指示灯
2).核心板发光二极管 D4 3.3V 电源指示灯
3).核心板发光二极管 D5 网线连接指示灯
4).核心板发光二极管 D6 网口数据读写指示灯
5).底板发光二极管 D11 3.3V 电源指示灯
6).底板发光二极管 D15 USB 读写指示灯
9).底板发光二极管 D1-D8 指示灯对应 GPE0-GPE7
10).底板跳线 J2 蜂鸣器使用跳线(加上短路帽使能蜂鸣器)

2.3 开发板接口使用说明

1).COM1 通用串行接口 1
2).COM2 通用串行接口 2
3).J15 液晶屏接口
4).RJ45 网口插座
5).SW1 电源开关
6).J9 电源插座
7).J10 立体声音频输出插座

8).JP3 A/D 采样输入接口及 GPE0-GPE7 外扩接口
9).JP2 20PIN 插座,标准 JTAG 接口
10).CON2 USB 从设备接口
11).MK1 蜂鸣器
12).S1-S4 键盘部分
13）.J1 USB 供电跳线（插上短路块，USB 供电，不插短路块，外接电源）
14）.J2 蜂鸣器使能跳线

2.4 开发板硬件资源说明

LJD-44B0XDVK-I 是核心板(4 层板)+底板(双面板)结构

核心板上资源：

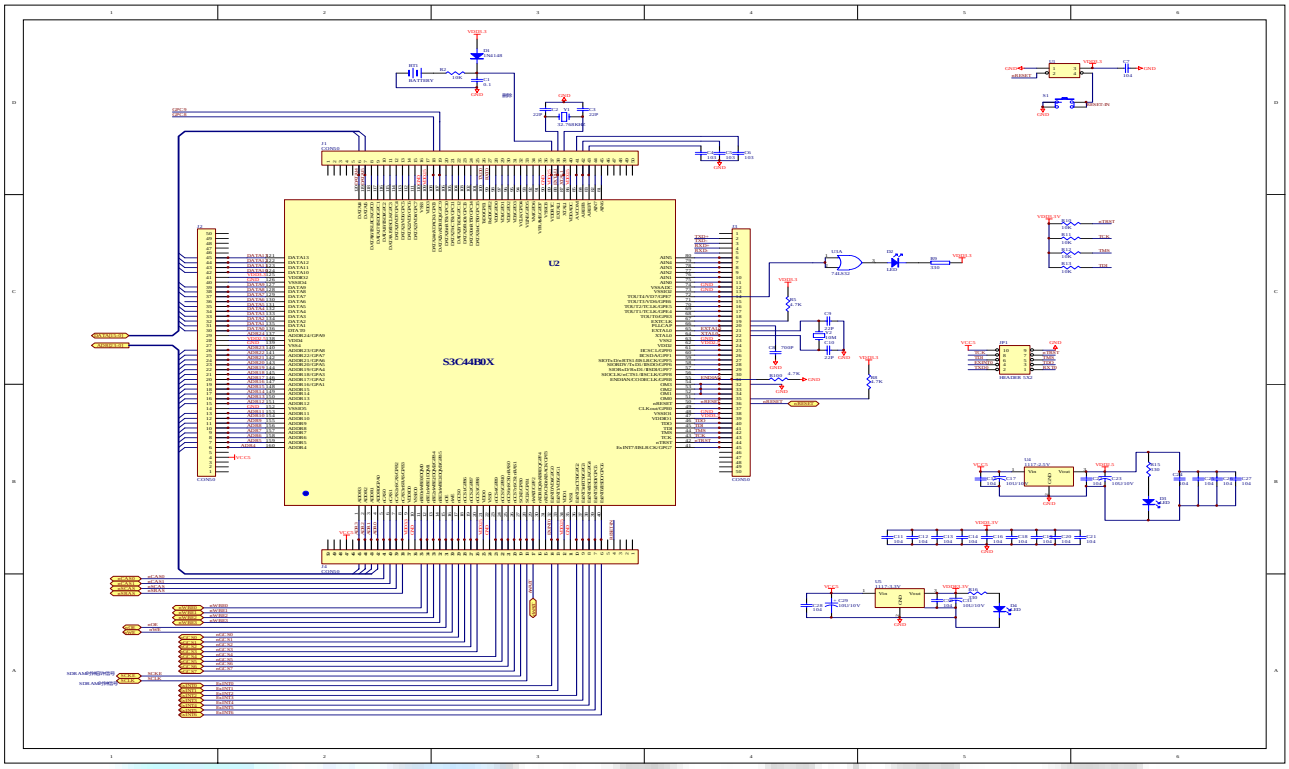
1).中央处理器 S3C44B0X（SAMSUNG），ARM7TDMI 外部存储器
2).2M Byte NOR FLASH（SST39VF1601，可扩展 4M 字节 SST39VF3201）；
3).8M Byte SDRAM （HY57V641620，可扩展 16M 字节的 SDRAM）；
4).可以扩展 16M /32M/64M/128M Byte NAND FLASH；
5).10M/100M 网口，CS8900；

底板上资源：

1).两个标准三线 RS232 接口（可扩展为 5 线）；
2).10M/100 网口，带发送和接收指示灯；
3).USB1.1 规范,PDIUSBD12，带联接状态指示灯；
4).LCD 接口信号线全部采用了 74HC244 进行了驱动，更加稳定可靠
5).支持单色、4 级灰度、16 级灰度、256 色 STN 液晶屏，最大支持 640x480/256 色 STN 液晶屏,可选配 16 级灰度屏 G35I、G35II、G57（带 触摸屏)或 256 色彩屏 C57(带触摸屏);
6).音频接口,带软件音量调节的双声道音频输出；
7).时钟源,内部实时时钟（带有后备锂电池）；
8).复位电路,一个复位按键，并采用专用复位芯片进行复位，稳定可靠；
9).调试及下载接口,一个 20 芯 Multi－ICE 标准 JTAG 接口，支持 SDT2.51,ADS1.2 等调试 ；
10).电源接口开关电源供电，输入直流电压范围是 5V，带电源指示灯；
11).4 个进口键盘，八个高亮 LED；
12).一个蜂鸣器（带使能控制的短路块）；
13).一个 EEPROM（AT24C02）用来验证 IIC 总线读写；
14).未使用的 ADC 引脚通过一个 10 芯标准连接器引出留给用户自己扩展；
15).操作系统 支持 uC/OS-II ，ucLinux ；

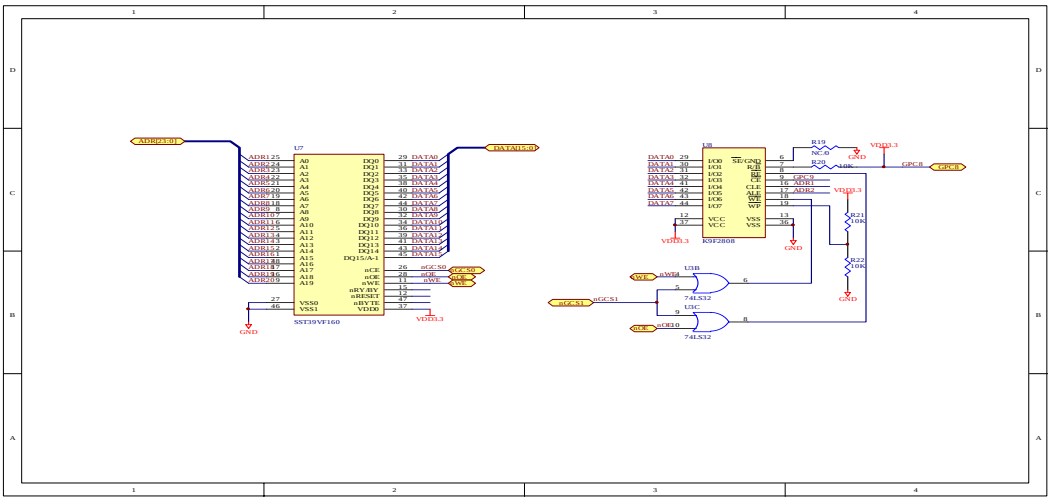
2.5 学习板核心板硬件电路说明

2.5.1 CPU 单元复位电路及电源电路



- CPU 及 I/O 口所需的电压由 LM117 3.3 供给
- CPU 所需的核电压由 LM1117-2.5V 供给
- 复位电路采用性能更稳定的专用复位芯片 MAX811 组成
- CPU 单元采用 16 位总线模式,外接晶体,采用内部倍频产生需要的频率

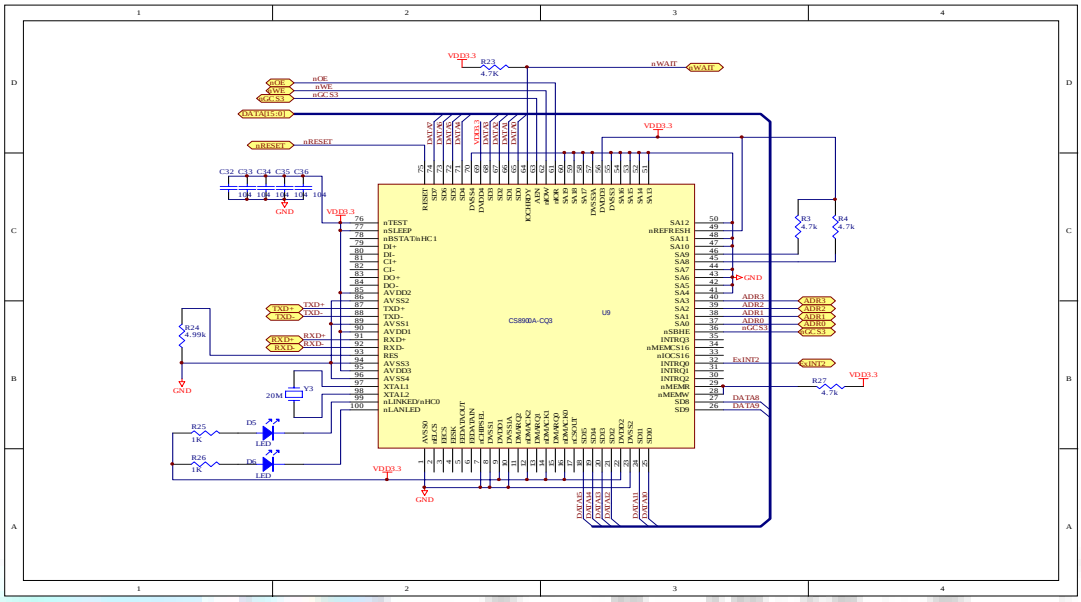
2.5.2 NORflash 和 nandflash 电路



◆ 由于大多数NAND FLASH 的引脚都是兼容的，所以这里的NAND FLASH 可以支持多种型号和容量

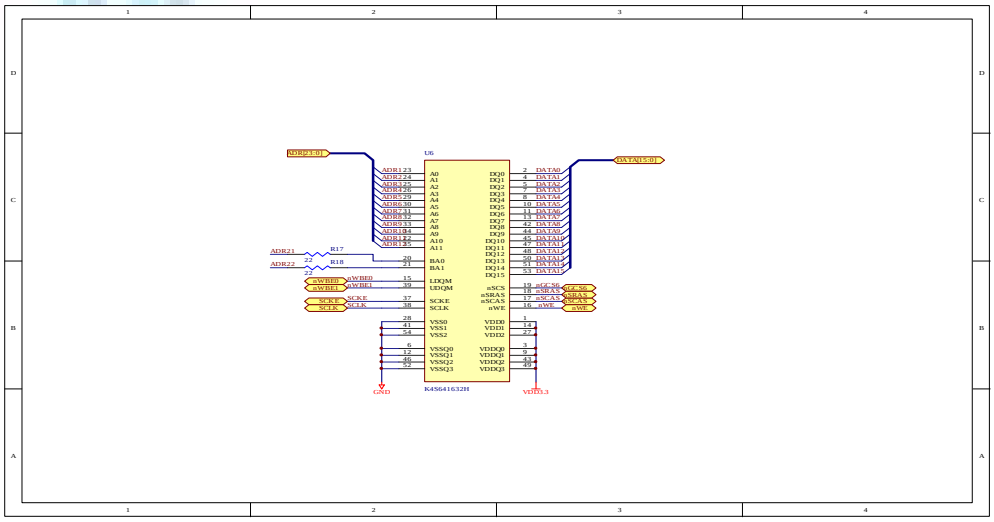
◆ 考虑到NOR FLASH 的货源和价格因素，NORFLASH 电路也采用了兼容设计方案，根据其周围焊接不同的电阻，可以支持多种不同型号与容量的NOR FLASH。

2.5.3 网卡接口芯片电路



LJD-44B0XDVK-I使用性能更好的8900A-CQ3 芯片扩展了网络通讯模块，它的传输速率工作为10M。在电路接口中，CS8900 工作在16 位模式下，网卡芯片复位默认工作方式为I/O 连接，寄存器的默认基址是300H。

2.5.4 SDRAM 电路



考虑到用户的实际需要，SDRAM 电路采用了兼容设计的方案，支持 8M 字节或 16M 字节 SDRAM，由四个电阻进行选择。如果是 8M 的 SDRAM（如 K4S641632）则只焊接电阻 R18 和 R30，R17 和 R31 悬空不用。如果焊的是 16M 的 SDRAM 则只焊接电阻 R17 和 R31,R18 和 R30 悬空不用。相关电路如上图：

第三章．开发板各个模块电路实验测试方法

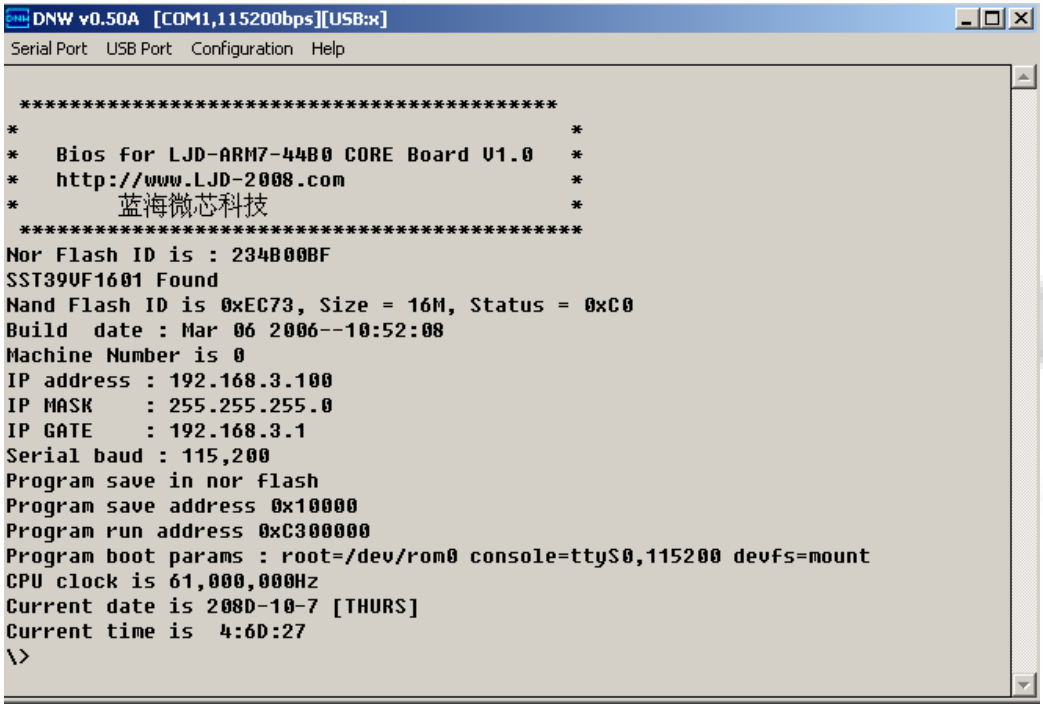
用户拿到开发板套件之后，请首先按照如下步骤进行测试，以验证开发板好坏。每一块开发板在发货之前，均烧写了 BIOS，设置好了初始化参数；

将 5V 电源接到板子上，用串口线、交叉网线将板子和 PC 机连接起来

在 PC 机上 DNW(0.50)，设置为：COM1，115200

3.1．如何运行测试程序

打开开发板上面的开关，上面的 LED 指示灯将会亮起。从 PC 机的 DNW 上面看到如下的信息：



上图信息是说明检测到了你的 PC 机网络设置为：

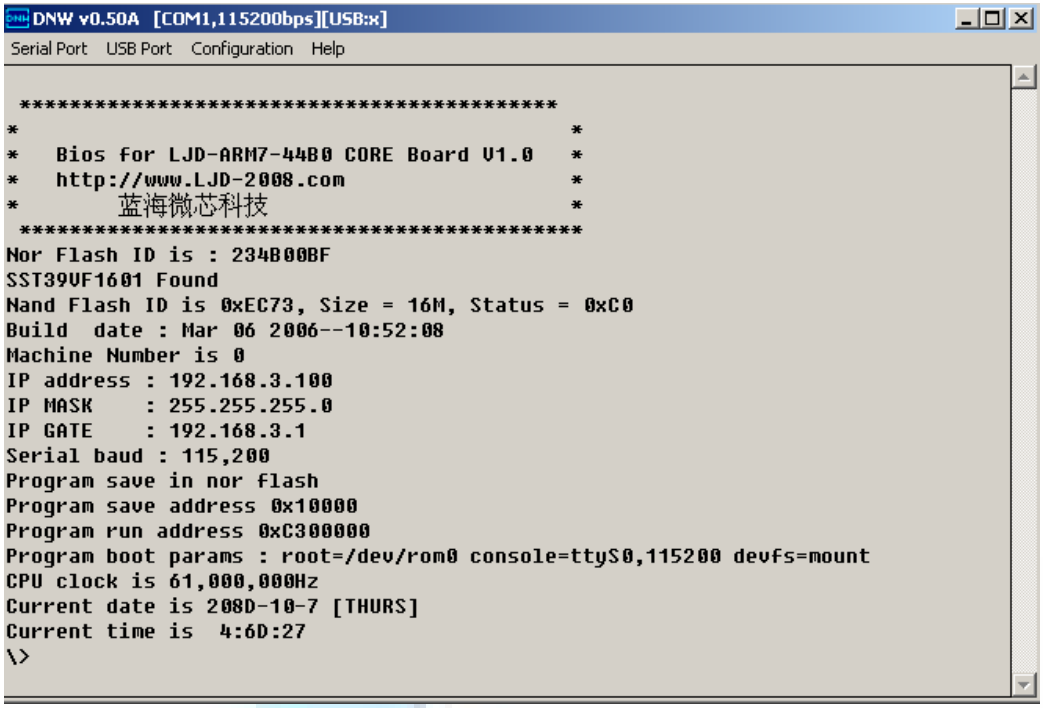
IP 地址：192.168.3.XXX（譬如设置 IP 为 192.168.3.100）

子网掩码：255.255.255.0

在 DNW 里面输入 BIOS 命令 netrun，启动 TFTP 协议准备从 PC 机上下载目标代码；

在 LJD-44B0DVK-I\目标代码目录下运行 LJD-44B0XDVK.BAT，将目标代码文件 LJD-44B0X.BIN 通过网口传到开发板 SDRAM 上从 0XC008000 开始的地方；

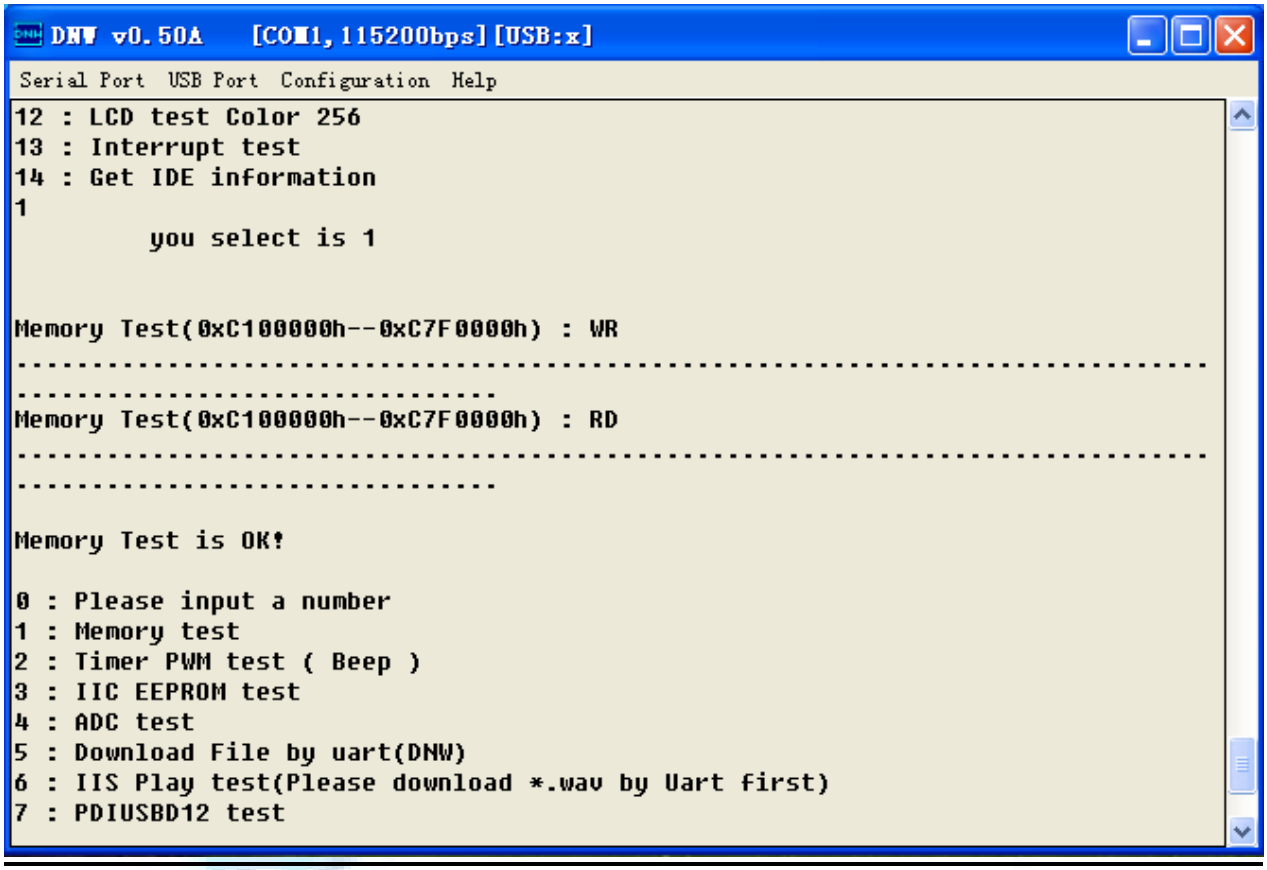
在 DNW 里面您将看到如下信息：



上面提示信息显示：IP 地址是 192.168.3.100，下载的代码放置在 SDRAM 中从 0xc300000 开始的地址，当下载结束后，会立即执行下载的程序，显示信息如下：

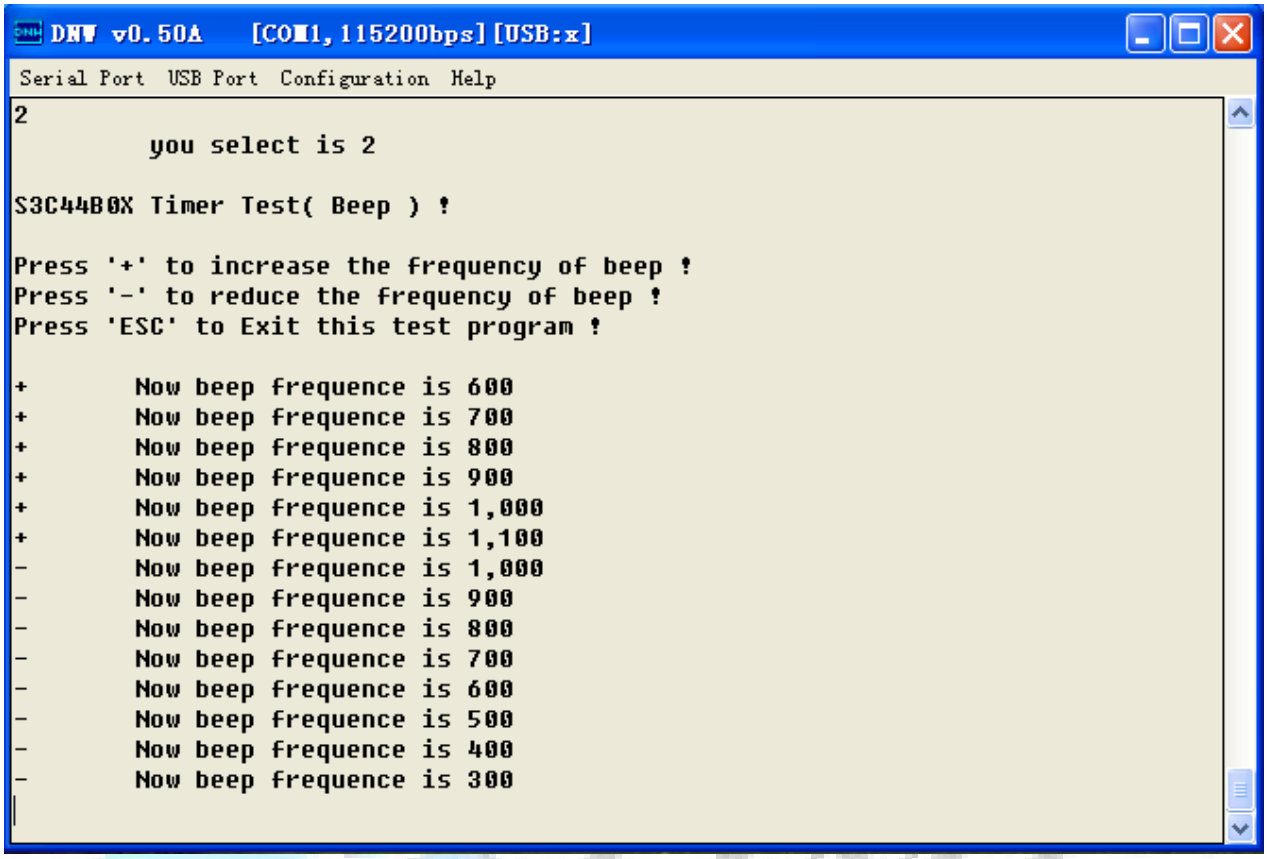
3.3 内存 SDRAM 读写自测试

按“1”然后回车进行内存（SDRAM）读写自测试，提示信息如下：



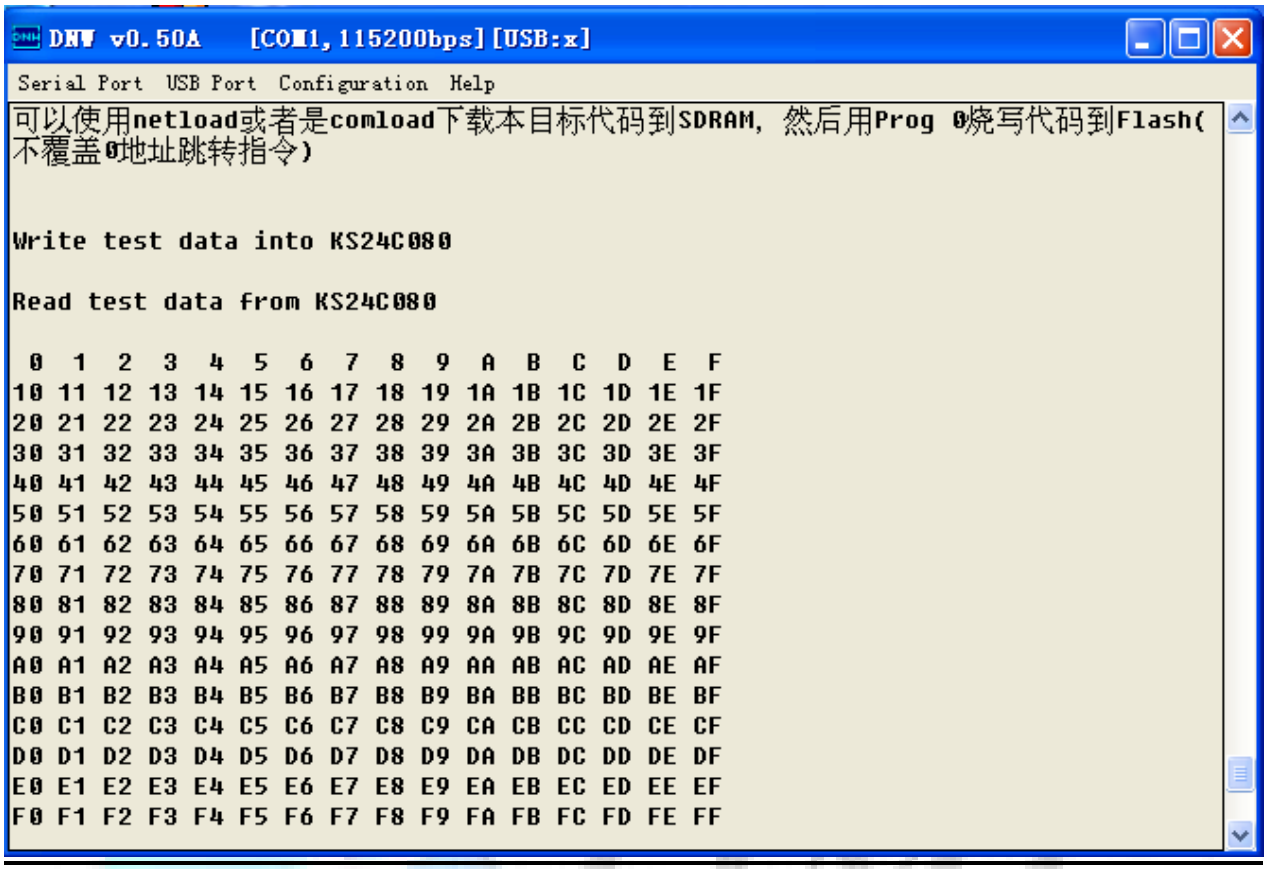
3.4 PWM 脉宽调试和蜂鸣器测试

按“2”然后回车测试 PWM（脉宽调制）输出并驱动一个蜂鸣器，注意 JP1 上一定要插上一个短路块，现在就可以听到蜂鸣器的鸣叫声了，按“＋”增加蜂鸣器的频率，按“－”减小蜂鸣器的频率，按“ESC”退出本测试返回到主测试菜单，提示信息如下：



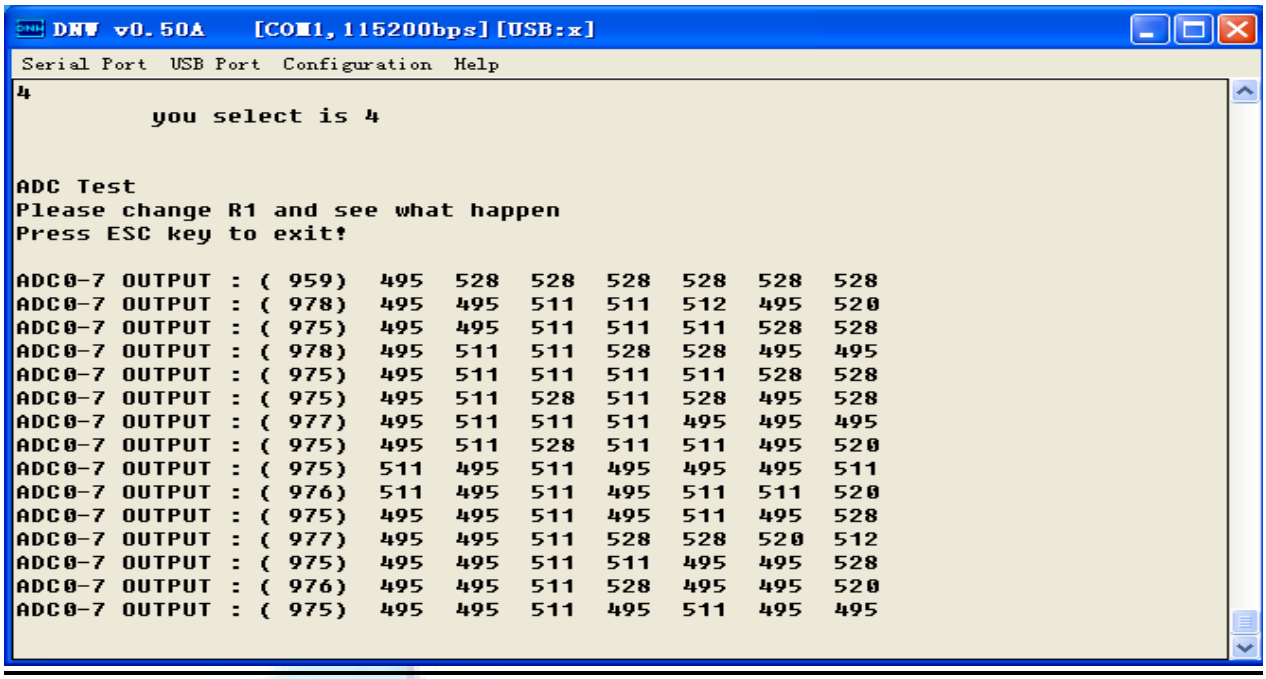
3.5 IIC 总线 EEPROM 读写测试

按“3”并回车测试 IIC 总线 EEPROM 读写，本测试使用了中断，将 EEPROM 中写入连续的数字再读出并显示出来。由于 DNW.EXE V0.50A 对中文支持不好，所以下面提示信息中的中文字符显示有乱码。这段中文的意思是如果要运行中断程序，必须将目标代码烧入到 Flash 中一次，因为 S3C44B0 的中断会首先跳转到 FLASH 的起始几个地址，然后在那里经跳转最后才到 SDRAM 中的中断服务程序。因为 BIOS 不支持中断，FLASH 的起始地址没有中断跳转向量表。所以只需要将本程序烧写到 FLASH 中一次，Flash 就有了中断向量表了。以后调试本程序或者是运行本程序，均不需要再烧写 FLASH，通过网口或者是串口下载到 SDRAM 中就可以运行中断了，提示信息如下：



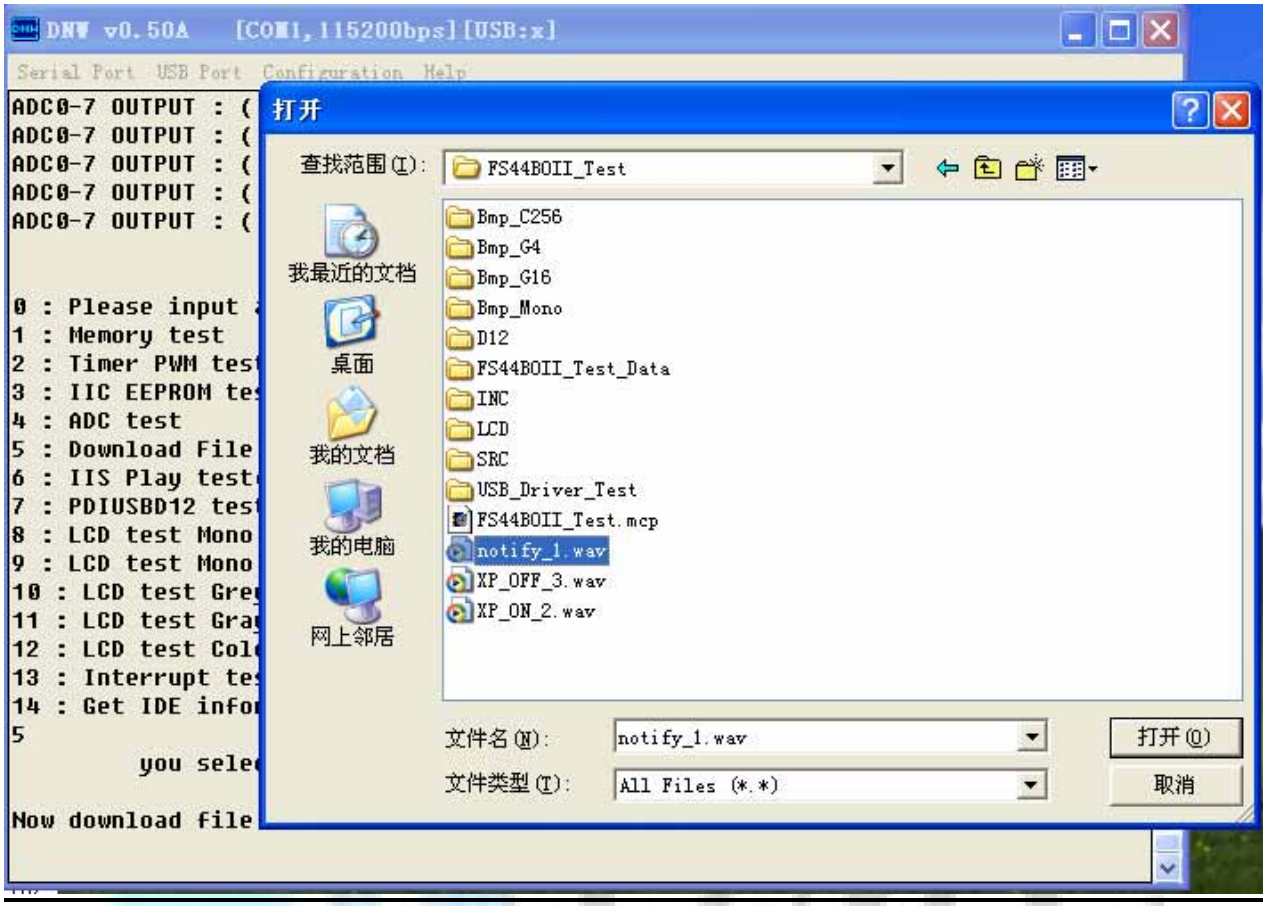
3.6 模数转换器 ADC 测试

按“4”并回车进行模数转换器（ADC）功能测试，因为 S3C44B0X 的模数转换器是 10 位精度的，提示信息如下：



3.7 IIS 音频播放 WAV 文件测试

第 6 项是测试 IIS 音频输出，不过在测试音频输出之前，首先需要通过串口下载一个 wav 格式的音频文件进来才可以。按“ 5 ”并回车开始下载*.WAV 文件到 SDRAM，然后点击 DNW.EXE 的菜单 SERIAL PORT/TRANSMIT，弹出一个文件浏览的窗口，在这窗口中选中光盘上面提供的 WAV 文件（譬如）NOTIFY.WAV，点击按键打开启动文件传输：



文件传输的时候，会出现传输的进度条，传输完毕后，会出现提示信息问您是否运行刚才下载的文件，这时要输入“N”不运行刚才下载的文件（因为 WAV 音频文件是不可以执行的）。

现在在主菜单里面按 “6” 并回车，把音箱的音频插头或者是耳机插头插入到开发板上面的音频插座，现在就可以听到刚才下载的 WAV 文件了，旋转音频插座左边的电位器调教音量，注意顺时针方向旋转是减小音量，逆时针方向旋转是增大音量，相关提示信息如下：

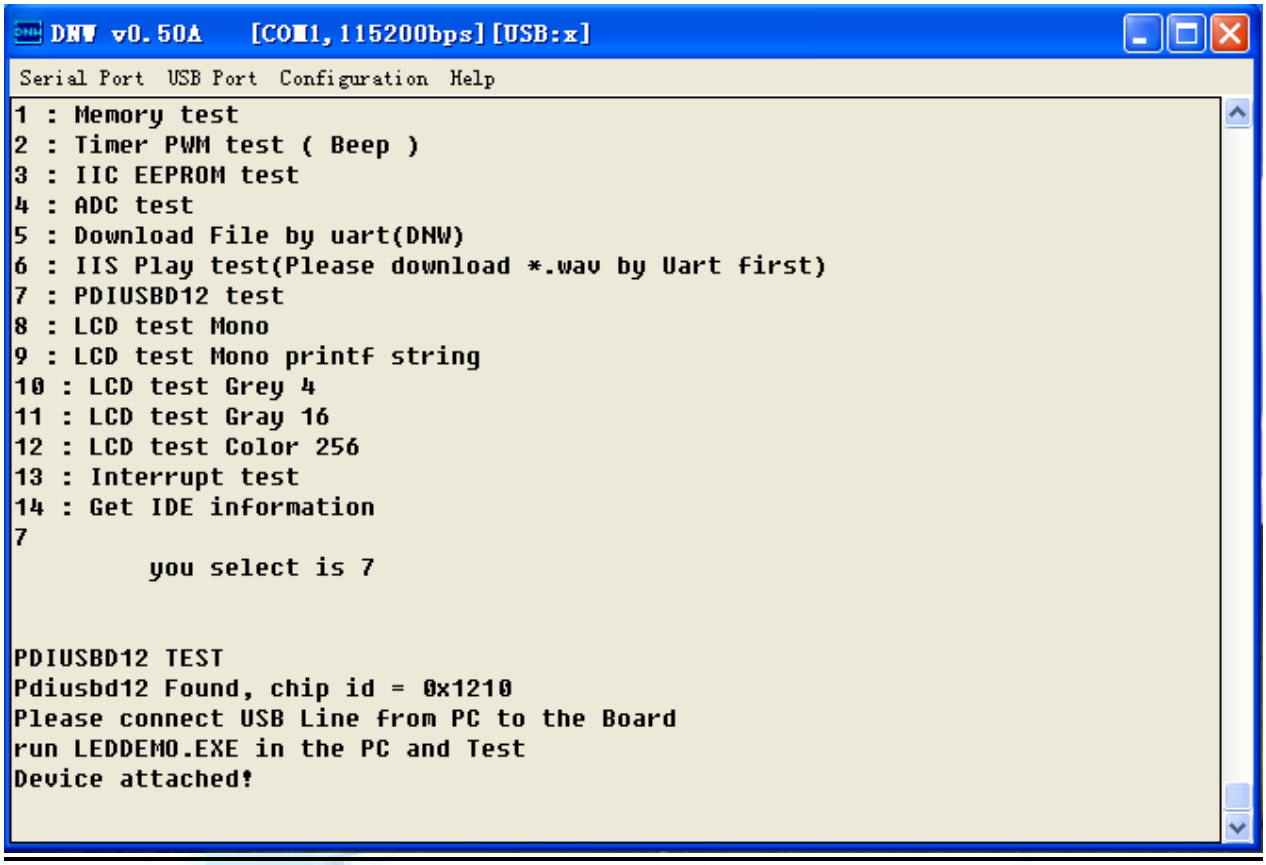
```
6
        you select is 6

Sample rate =22,050Hz

sample start:0xC40002C
sample size:0x1D1FC
Now play the wave file ...
Push ESC key to exit!!!
##########################################
```

3.8 USB DEVICE（PDIUSBD12）测试

按“ESC”退出前面的测试，按“7”测试 USB DEVICE（PDIUSBD12），用 USB 线连接 PC 机和目标板，可以看到下面的提示信息：


DNW v0.50A [COM1,115200bps][USB:x]
Serial Port USB Port Configuration Help
1 : Memory test
2 : Timer PWM test (Beep)
3 : IIC EEPROM test
4 : ADC test
5 : Download File by uart(DNW)
6 : IIS Play test(Please download *.wav by Uart first)
7 : PDIUSBD12 test
8 : LCD test Mono
9 : LCD test Mono printf string
10 : LCD test Grey 4
11 : LCD test Gray 16
12 : LCD test Color 256
13 : Interrupt test
14 : Get IDE information
7
you select is 7
PDIUSBD12 TEST
Pdiusbd12 Found, chip id = 0x1210
Please connect USB Line from PC to the Board
run LEDDEMO.EXE in the PC and Test
Device attached!

在 PC 机里面运行所提供的测试程序 LEDDEMO.EXE（注意必须先装好驱动程序 LJD-44B0DVK-I\DEMO\LJD-44B0XDVK_Test \USB_DRIVER_TEST\D12 驱动），点击三个按键可以看到开发板上面 LED 分别亮和灭。

3.9 黑白 STN 液晶屏测试

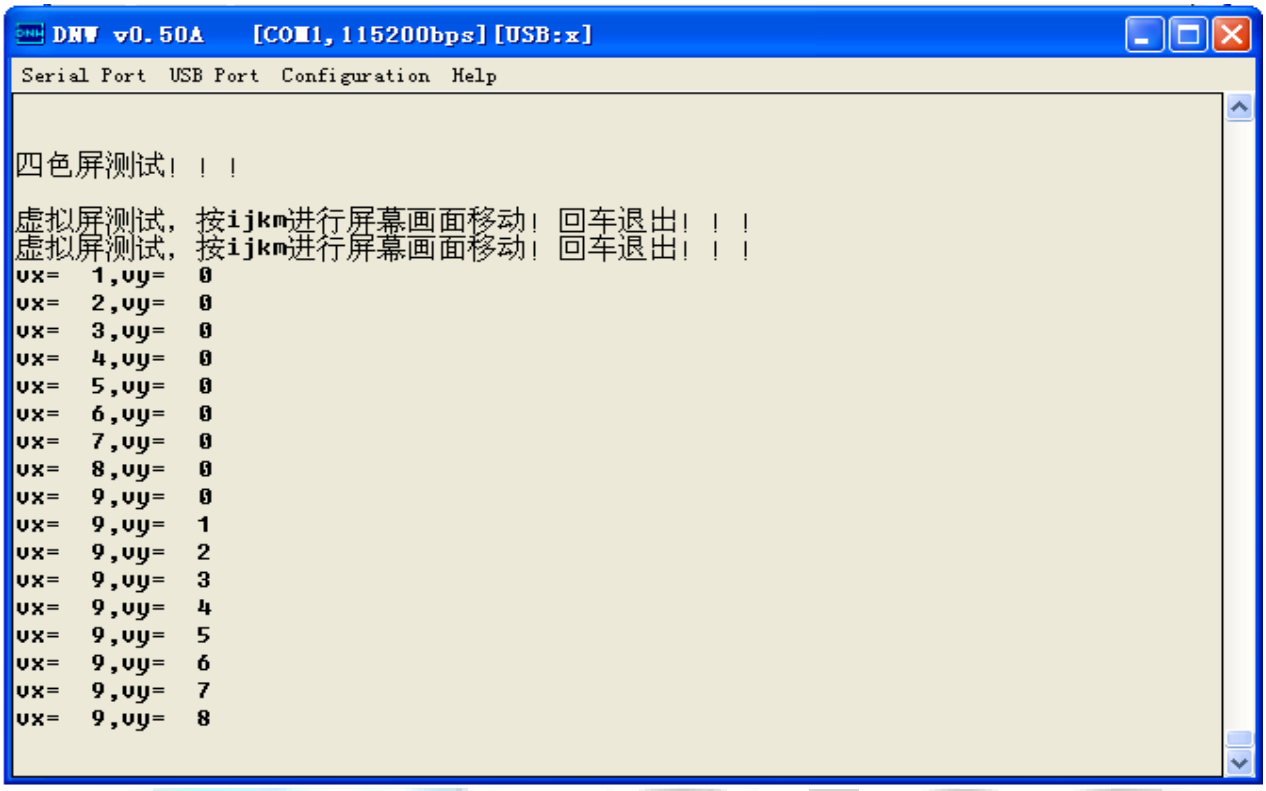
按“ESC”退出 PDIUSBD12 的测试，按“8”测试黑白的 STN 液晶屏，接上 G35（16 级灰度）或者是 G57（黑白）的 STN 液晶屏，按照提示信息按回车或者是“i”、“j”、“k”，“m”，可以看到液晶屏上面出现不同的画面，相关提示信息如下：

3.10 黑白 STN 液晶屏显示英文字符测试

按“9”进行黑白 STN 液晶屏输出字符串测试，接上 G35（16 级灰度）或者是 G57（黑白）的 STN 液晶屏，按照提示信息按回车或者是“i”、“j”、“k”，“m”，可以看到液晶屏上面出现不同的画面，相关提示信息如下：

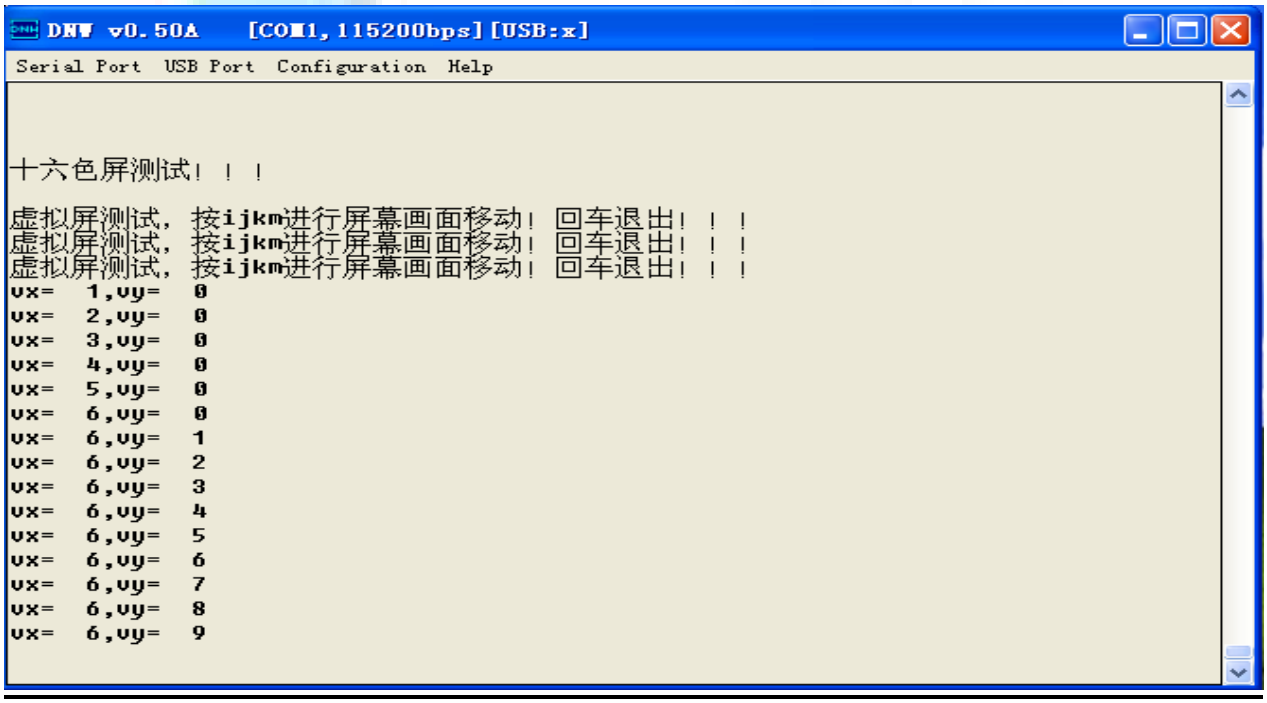
3.11 4 级灰度 STN 液晶屏测试

按“10”进行 4 级灰度 STN 液晶屏的测试，接上 G35（16 级灰度）或者是 G57（黑白）的 STN 液晶屏，按照提示信息按回车或者是“i”、“j”、“k”，“m”，可以看到液晶屏上面出现不同的画面，相关提示信息如下：



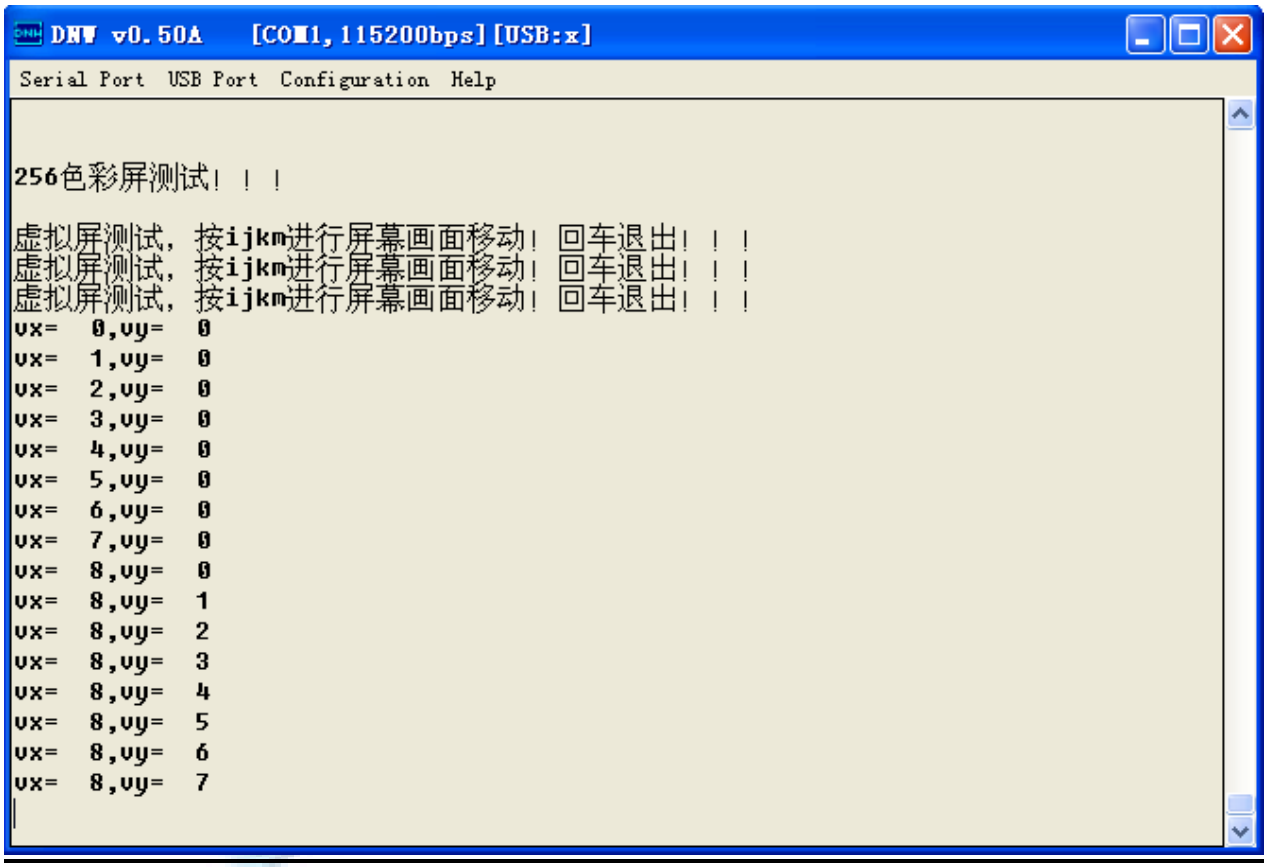
3.12 16 级灰度 STN 液晶屏测试

按“11”进行 16 级灰度 STN 液晶屏的测试，接上 G35（16 级灰度）或者是 G57（黑白）的 STN 液晶屏，按照提示信息按回车或者是“i”、“j”、“k”，“m”，可以看到液晶屏上面出现不同的画面，相关提示信息如下：



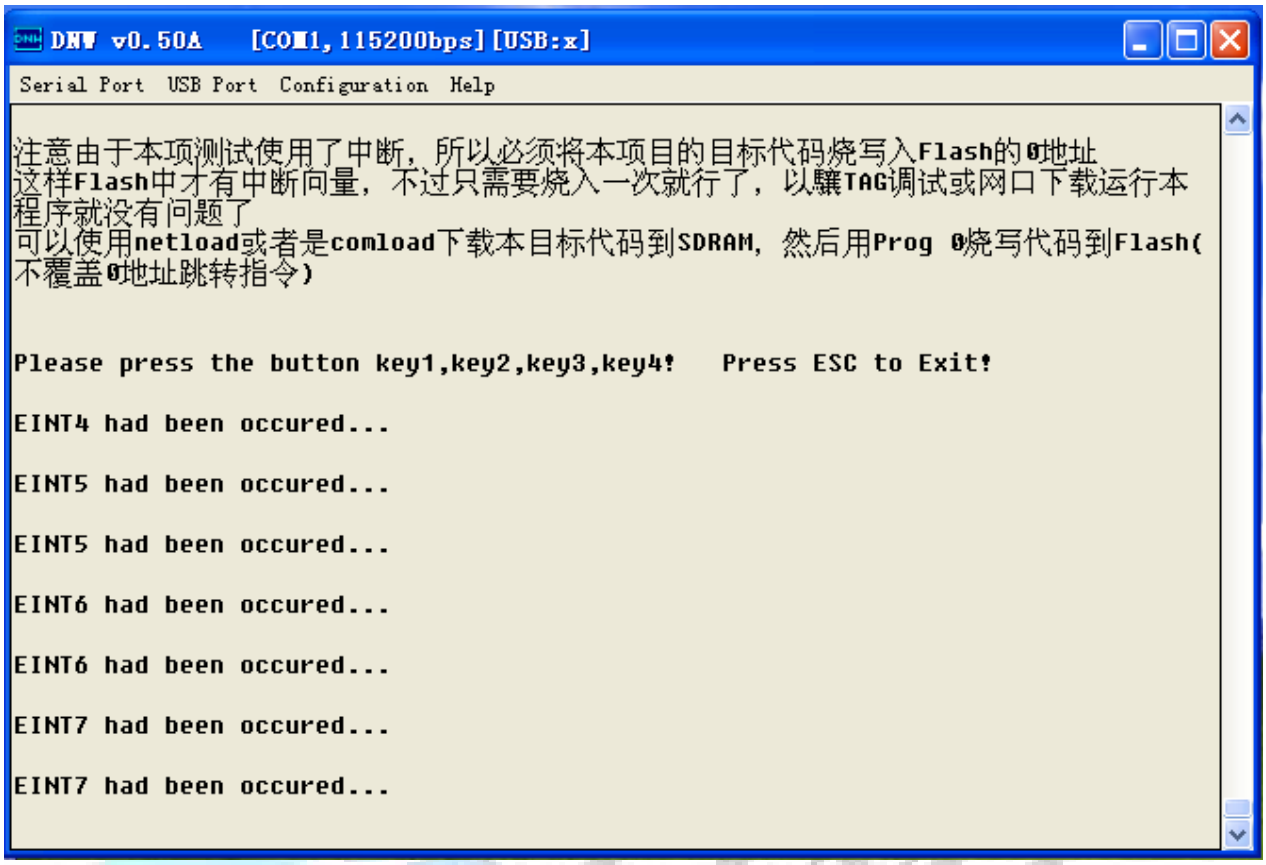
3.13 256 色 STN 液晶屏测试

按“12”进行 256 色 STN 液晶屏的测试，接上 G35（16 级灰度）或者是 G57（黑白）的 STN 液晶屏，按照提示信息按回车或者是“i”、“j”、“k”，“m”，可以看到液晶屏上面出现不同的画面，相关提示信息如下：



3.14 外部电平中断测试

按“13”测试外部电平中断，按板子上面的按键 S3、S4，就可以看到中断响应信息，相关提示信息如下：



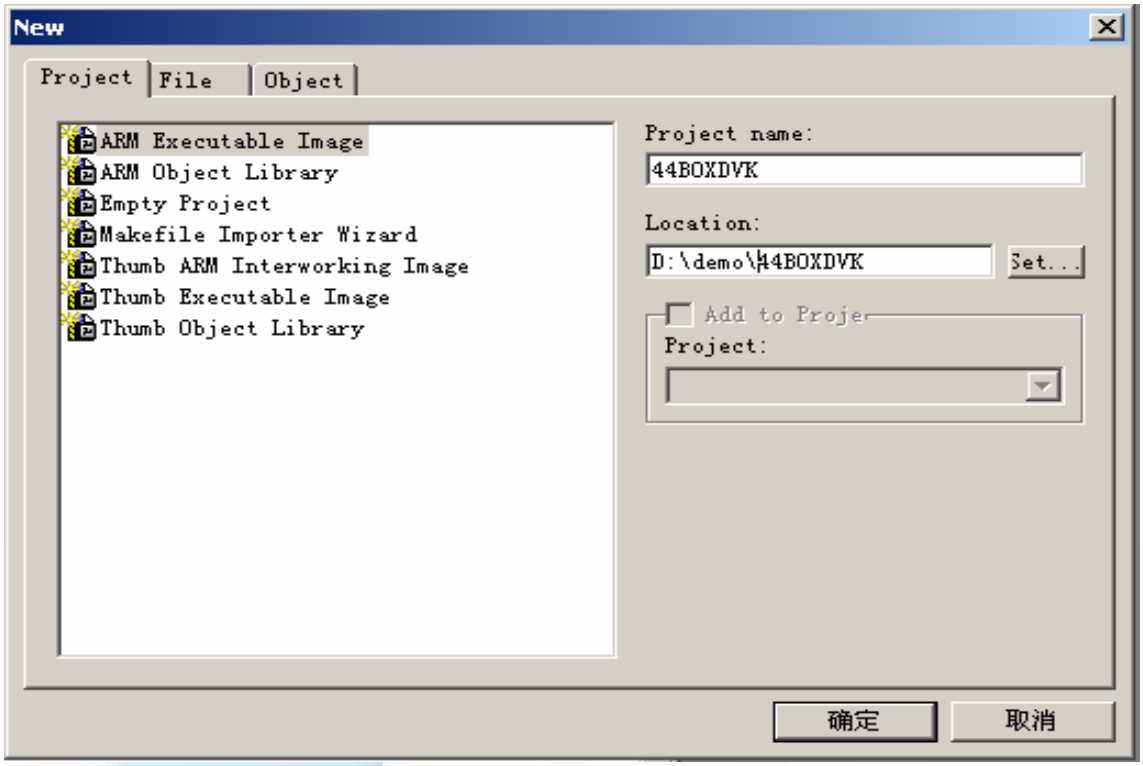
第四章．ADS 开发环境使用介绍

4.1 使用 ADS 创建工程

本节通过一个具体实例，为读者介绍如何使用该集成开发环境，利用CodeWarrior 提供的建立工程的模板建立自己的工程，并学会如何进行编译链接，生成包含调试信息的映像文件和可以直接烧写的FLASH 中的.bin 格式的二进制可执行文件。

4.1.1 建立一个工程

工程将所有的源码文件组织在一起，并能够决定最终生成文件存放的路径，输出的格式等。在CodeWarrior 中新建一个工程的方法有两种，可以在工具栏中单击“ New ”按钮，也可以在“ File ”菜单中选择“ New… ”菜单。会打开一个如图(4.1-1)所示的对话框。



新建工程对话框在这个对话框中为用户提供了 7 种可选择的工程类型。

ARM Executabl Image：用于由ARM 指令的代码生成一个ELF 格式的可执行映像文件；

ARM Object Library：用于由ARM 指令的代码生成一个armar 格式的目标文件库；

Empty Project：用于创建一个不包含任何库或源文件的工程；

Makefile Importer Wizard：用于将Visual C 的nmake 或GNU make 文件转入到CodeWarrior IDE工程文件；

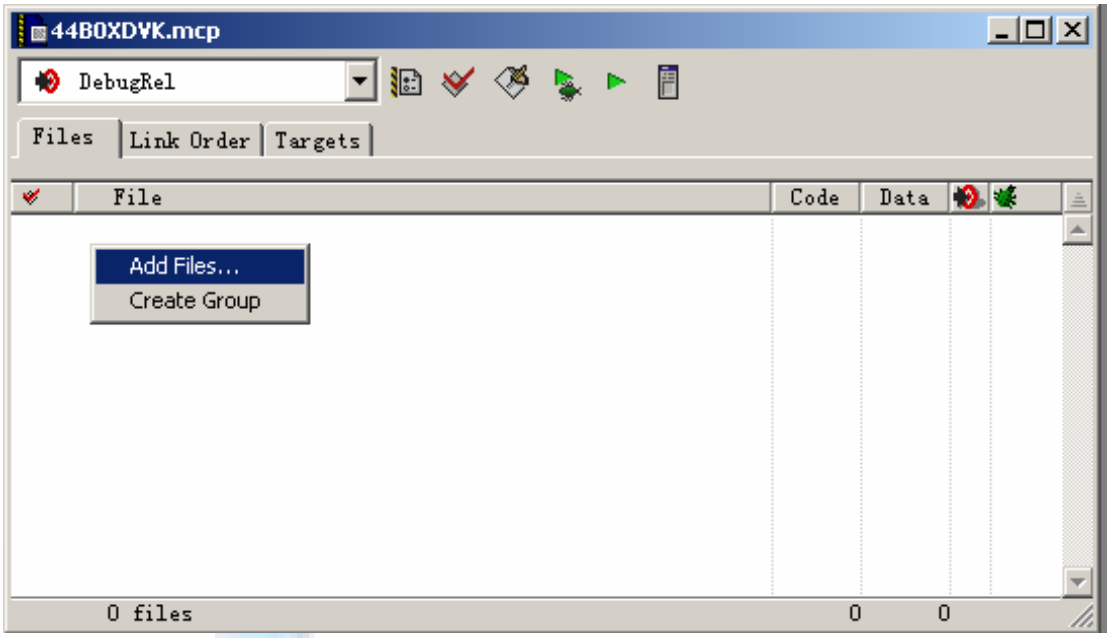
Thumb ARM Executable Image：用于由ARM 指令和Thumb 指令的混和代码生成一个可执行的ELF 格式的映像文件；

Thumb Executable image：用于由Thumb 指令创建一个可执行的ELF 格式的映像文件；

Thumb Object Library：用于由Thumb 指令的代码生成一个armar 格式的目标文件库。

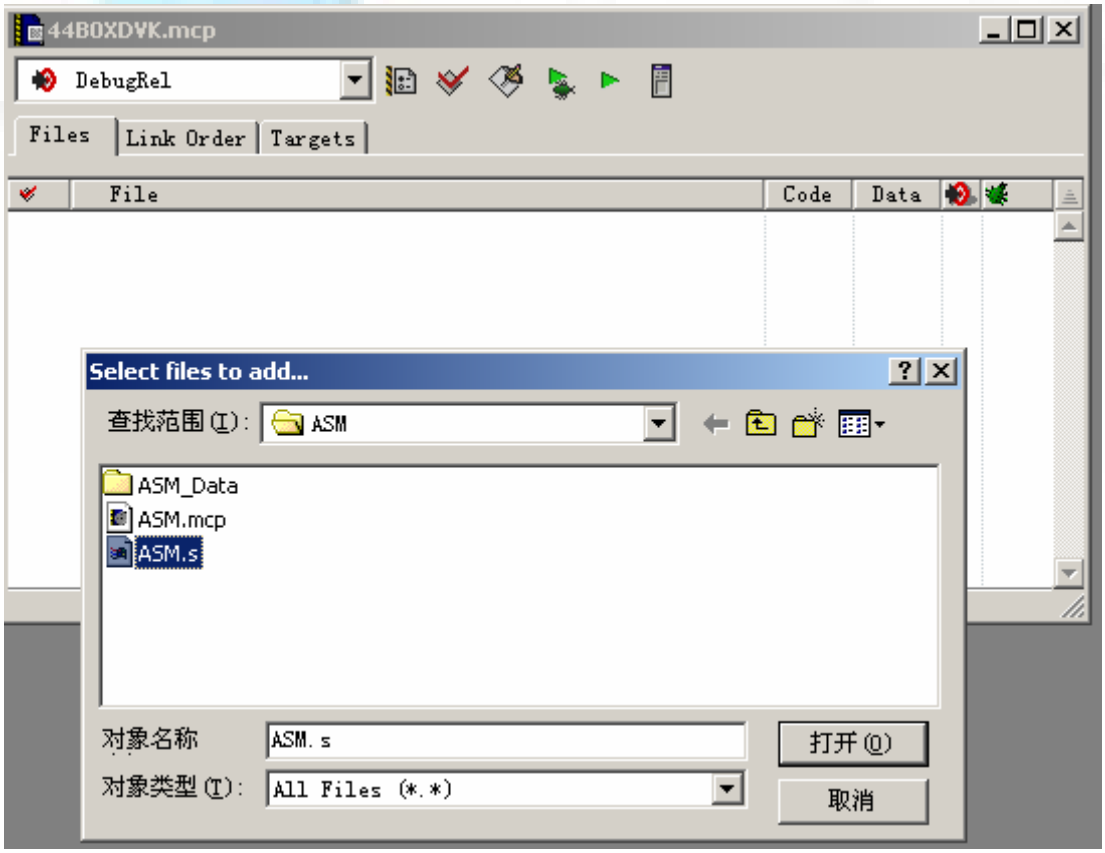
在这里选择ARM Executable Image，在“Project name:”中输入工程文件名，本例为“44B0XDVK”，点击“Location:”文本框的“Set…”按钮，浏览选择想要将该工程保存的路径，将这些设置好后，点击“确定”，即可建立一个新的名为44B0XDVK 的工程。

这个时候会出现44B0XDVK.mcp 的窗口，如图4.1-2 所示，有三个标签页，分别为files,link order,targets 默认的是显示第一个标签页files。通过在该标签页点击鼠标右键，选中“AddFiles…”可以把要用到的源程序添加到工程中。



新建工程打开窗口

对于本例，将已准备好的源文件（.s 和.c 文件）添加进去，添加过程如下图(图4.1-3)：



点击“打开”按钮确定。

在这里还有一个细节，希望读者注意。在建立好一个工程时，默认的target 是DebugRel，还有另外两个可用的target，分别为Realse 和Debug，这三个target 的含义分别为：

DebugRel：使用该目标，在生成目标的时候，会为每一个源文件生成调试信息；

Debug：使用该目标为每一个源文件生成最完全的调试信息；

Release：使用该目标不会生成任何调试信息。

在本例中，使用默认的DebugRel 目标。

现在已经新建了两个源文件，要把这两个源文件添加到工程中去。

为工程添加源码常用的方法有两种，既可以使用入图4.1-3 所示方法，也可以“ Project ”菜单项中，选择“ Add Files… ”，这两种方法都会打开文件浏览框，用户可以把已经存在的文件添加到工程中来。当选中要添加的文件时，会出现一个对话框，如图

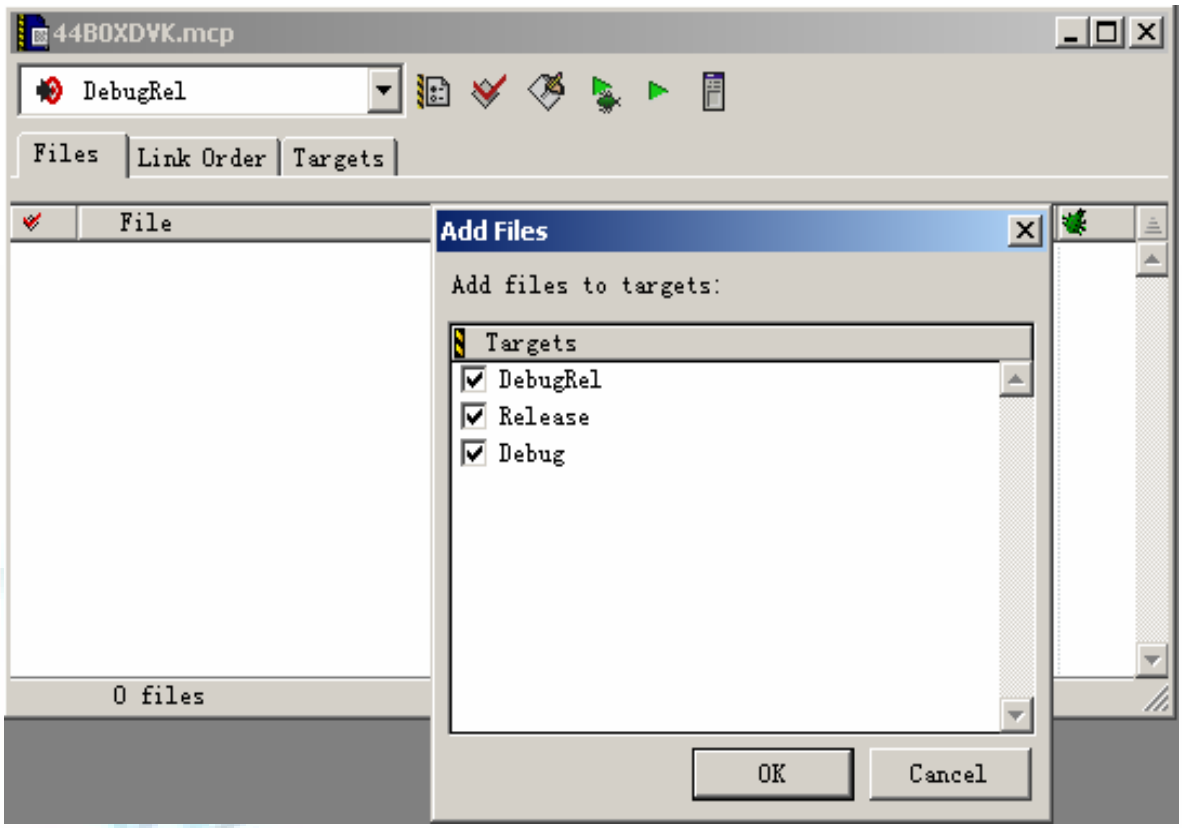


图 4.1-4

4.1-4 所示，询问用户把文件添加到何类目标中，在这里，我们选择DebugRel 目标。把刚才创建的两个文件添加到工程中来。

到目前为止，一个完整的工程已经建立。下面该对工程进行编译和链接工作。

4.1.2 编译和连接工程

在进行编译和链接前，首先讲述一下如何进行生成目标的配置。

点击Edit 菜单，选择“ DebugRel Settings… ”(注意，这个选项会因用户选择的不同目标而有所不同)，出现如图4.1.5 所示的对话框。

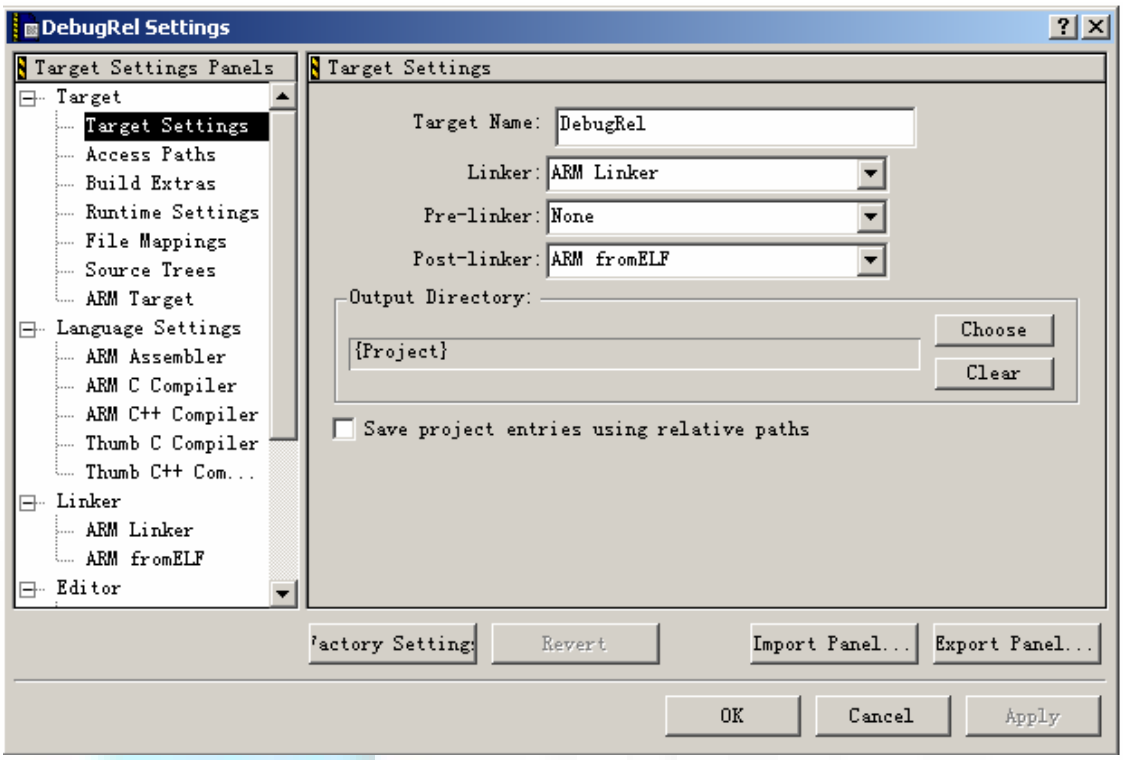


图4.1.5 DebugRel 设置对话框

这个设置中还有两个可选项，None 不是不用任何链接器，如果使用它，则工程中的所有文件都不会被编译器或汇编器处理。ARM Librarian 表示将编译或汇编得到的目标文件转换为ARM 库文件。对于本例，使用默认的链接器ARM Linker。

Pre-linker：目前CodeWarrior IDE 不支持该选项。

Post-Linker：选择在链接完成后，还要对输出文件进行的操作。因为在本例中，希望生成一个可以烧写到Flash 中去的二进制代码，所以在这里选择ARM fromELF，表示在链接生成映像文件后，再调用FromELF 命令将含有调试信息的ELF 格式的映像文件转换成其他格式的文件。

◆ Language Settings

因为本例中包含有汇编源代码，所以要用到汇编器。首先看ARM 汇编器。这个汇编器实际就说armasm，默认的ARM体系结构是ARM7TDMI，正好符合目标板S3C44b0x，无需改动。字节顺序默认就是小端模式。其他设置，就用默认值即可。

还有一个需要注意的就是ARM C 编译器，它实际就是调用的命令行工具armcc。使用默认的设置就可以了。细心的读者可能会注意到，在设置框的右下脚，当对某项设置进行了修改，该行中的某个选项就会发生相应的改动。实际上，这行文字就显示的是相应的编译或链接选项，由于有了CodeWarrior，开发人员可以不用再去查看繁多的命令行选项，只要在界面中选中或撤消某个选项，软件就会自动生成相应的代码，为不习惯在DOS 下键入命令行的用户提供了极大的方便。

◆Linker 设置

鼠标选中ARM Linker，出现如图4.1-6 所示对话框。这里详细介绍该对话框的主要的标签页选项，因为这些选项对最终生成的文件有着直接的影响。

在标签页Output 中，Linktype 中提供了三种链接方式。Partial 方式表示链接器只进行部分链接，经过部分链接生成的目标文件，可以作为以后进一步链接时的输入文件。Simple 方式是默认的链接方式，也是最为频繁使用的链接方式，它链接生成简单的ELF 格式的目标文件，使用的是链接器选项中指定的地

址映射方式。Scattered 方式使得链接器要根据scatter 格式文件中指定的地址映射，生成复杂的ELF 格式的映像文件。这个选项一般情况

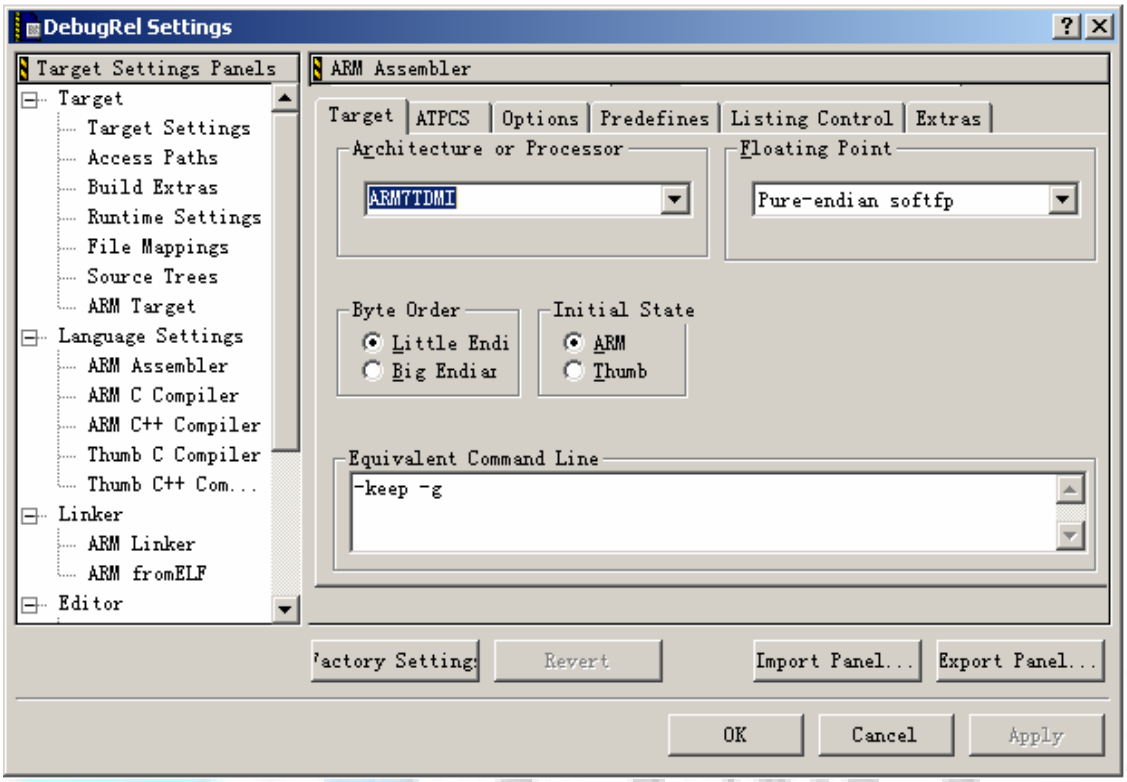


图 4.1-6 命令行工具选项设置

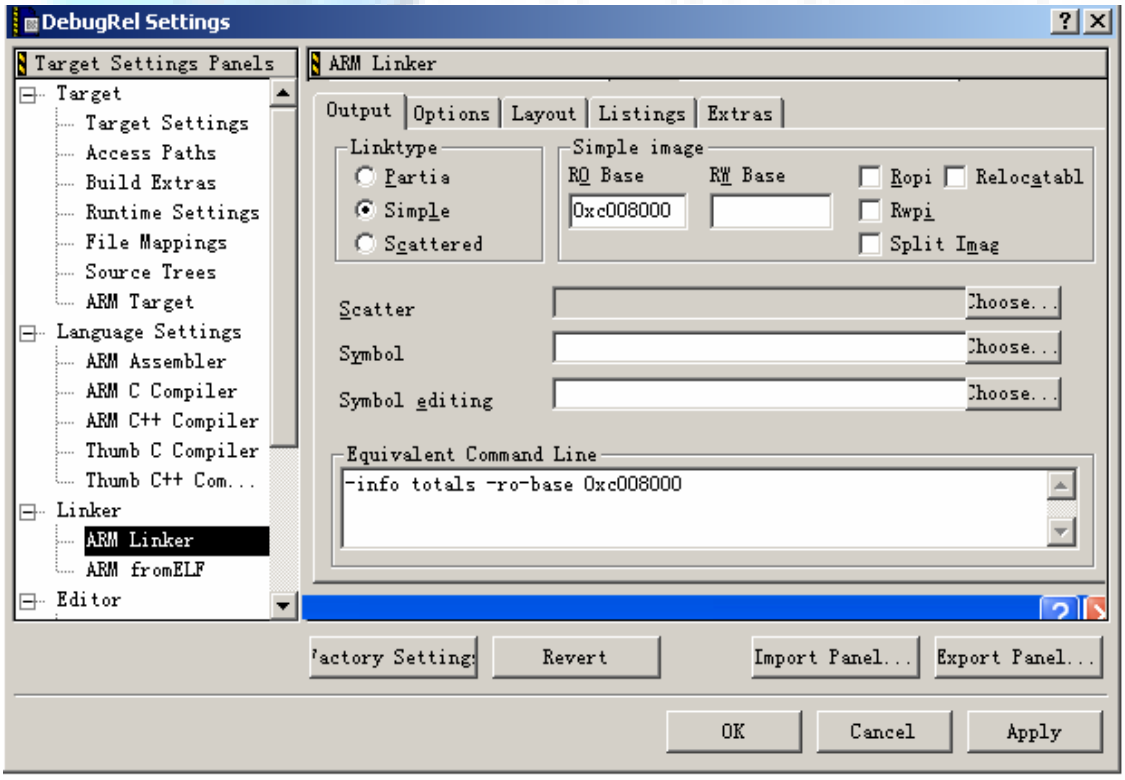


图4.1-7 链接器设置1

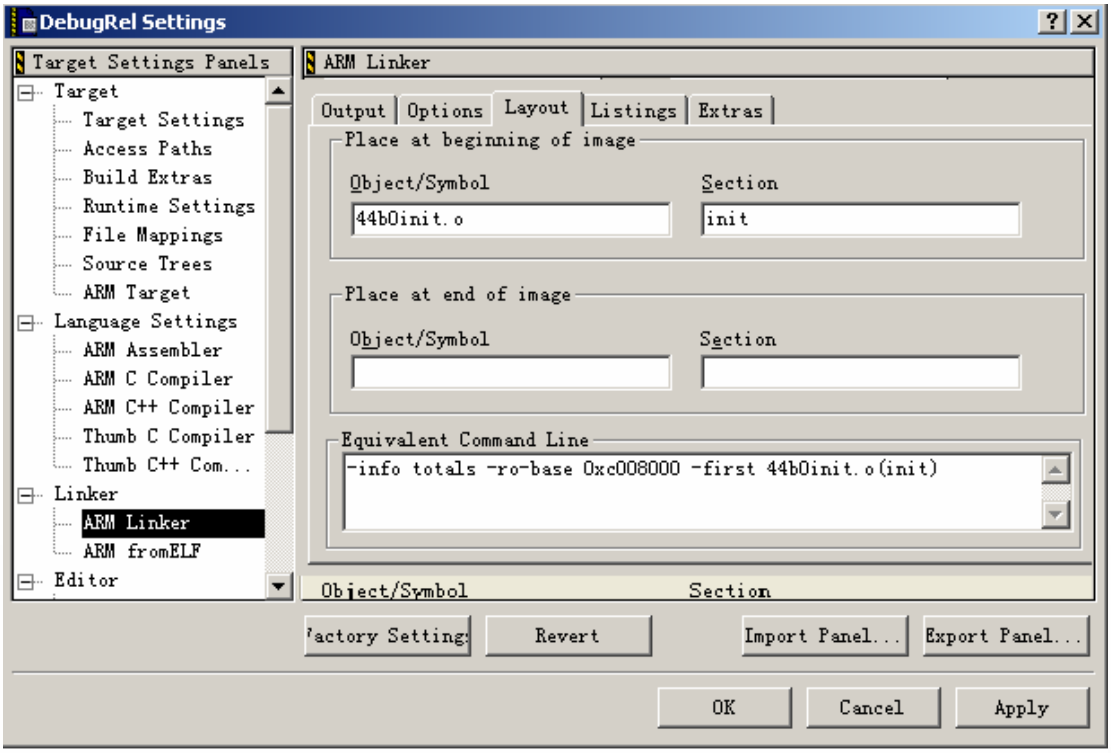


图4.1-8 设置映像文件的入口点

下，使用不太多。

因为所举的例子比较简单，选择Simple 方式就可以了。

在选中Simple 方式后，就会出现Simple image。

RO Base：这个文本框设置包含有RO 段的加载域和运行域为同一个地址。默认是0x8000。这里用户要根据自己硬件的实际SDRAM 的地址空间来修改这个地址，保证在这里填写的地址，是程序运行时，SDRAM 地址空间所能覆盖的地址。针对本书所介绍的目标板，就可以使用这个默认地址值。

RW Base：这个文本框设置了包含RW 和ZI 输出段的运行域地址。如果选中split 选项，链接器生成的映像文件将包含两个加载域和两个运行域，此时，在RW Base 中所输入的地址为包含RW和ZI 输出段的域设置了加载域和运行域地址

Ropi：选中这个设置将告诉链接器使包含有RO 输出段的运行域位置无关。使用这个选项，链接器将保证下面的操作：

检查各段之间的重定址是否有效；

确保任何由armlink 自身生成的代码是只读位置无关的。

Rwpi：选中该选项将会告诉链接器使包含RW 和ZI 输出段的运行域位置无关。如果这个选项没有被选中，域就标识为绝对。每一个可写的输入段必须是读写位置无关的。如果这个选项被选中，链接器将进行下面的操作，检查可读/可写属性的运行域的输入段是否设置了位置无关属性；检查在各段之间的重地址是否有效；

在Region$$Table 和ZISection$$Table 中添加基于静态存储器sb 的选项。

该选项要求RW Base 有值，如果没有给它指定数值的话，默认为0 值。

Split Image：选择这个选项把包含RO 和RW 的输出段的加载域分成2 个加载域：一个是包含RO 输出段的域，一个是包含RW 输出段的域。

这个选项要求RW Base 有值，如果没有给RW Base 选项设置，则默认是-RW Base 0。

Relocatable：选择这个选项保留了映像文件的重定址偏移量。这些偏移量为程序加载器提供了有用

信息。

在Options 选项中，需要读者引起注意的是Image entry point 文本框。它指定映像文件的初始入口点地址值，当映像文件被加载程序加载时，加载程序会跳转到该地址处执行。如果需要，用户可以在这个文本框中输入下面格式的入口点：

入口点地址：这是一个数值，例如-entry 0x0

符号：该选项指定映像文件的入口点为该符号所代表的地址处，比如：-entry int_handler如果该符号有多处定义存在，armlink 将产生出错信息。

offset+object(section)：该选项指定在某个目标文件的段的内部的某个偏移量处为映像文件的入口地址，例如：-entry 8+startup(startupseg)在此处指定的入口点用于设置ELF 映像文件的入口地址。需要引起注意的是,这里不可以用符号main 作为入口点地址符号,否则将会出现类似“ Image dose not have an entry point(Not specified or not set due to multiple choice) ”的错误信息。在**Layout** 选项中，需要的设置如图**4.1-8**，设置**44B0INIT.O** 目标文件中的**Init** 为整个文件的入口点。

关于ARM Linker 的设置还有很多，对于想进一步深入了解的读者，可以查看帮助文件，都有很详细的介绍。在Linker 下还有一个ARM fromELF，如图8.7 所示：

fromELF 是一个实用工具，它实现将链接器，编译器或汇编器的输出代码进行格式转换的功能。例如，将ELF 格式的可执行映像文件转换成可以烧写到ROM 的二进制格式文件；对输出文件进行反汇编，从而提取出有关目标文件的大小，符号和字符串表以及重定址等信息。

只有在Target 设置中选择了Post-linker，才可以使用该选项。

在Output format 下拉框中，为用户提供了多种可以转换的目标格式，本例选择Plain binary，这是一个二进制格式的可执行文件，可以被烧些的目标板的Flash 中。在Output file name 文本域输入期望生成的输出文件存放的路径，或通过点击Choose...按钮从文件对话框中选择输出文件。如果在这个文本域不输入路径名，则生成的二进制文件存放在工程所在的目录下。

进行好这些相关的设置后，以后在对工程进行make 的时候，CodeWarrior IDE 就会在链接完成后调用fromELF 来处理生成的映像文件。对于本例的工程而言，到此，就完成了make 之前的设置工作了。

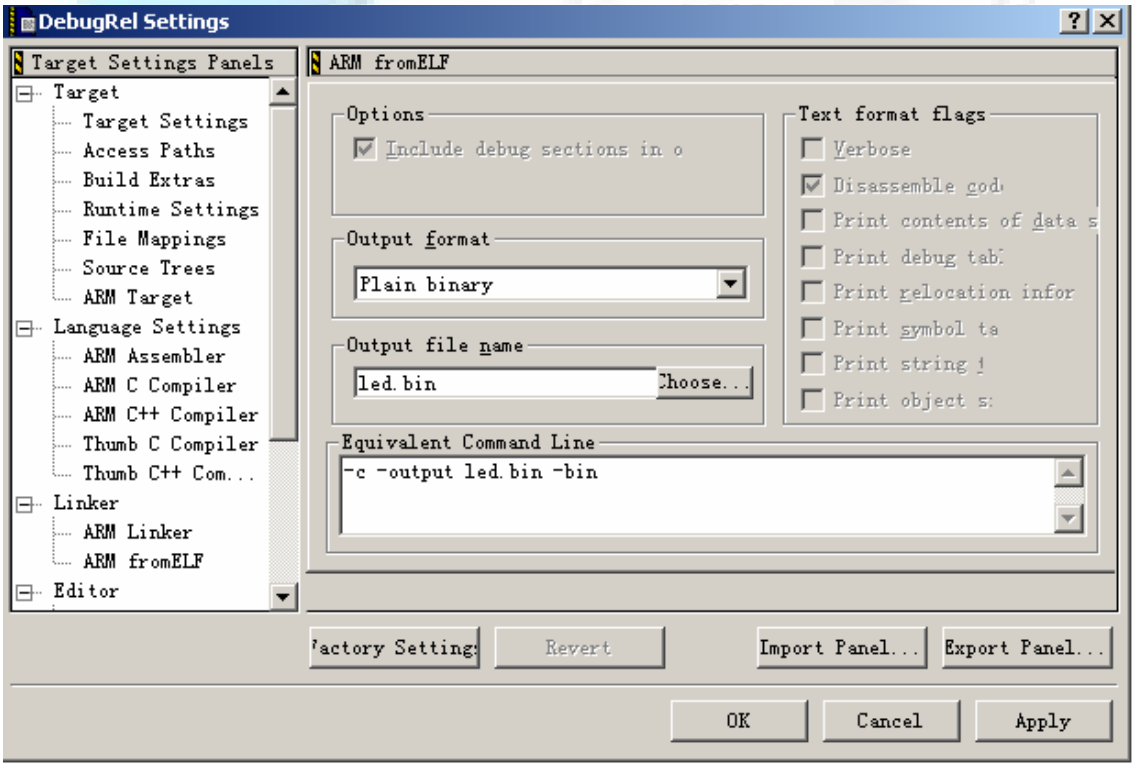


图4.1-9 ARM fromELF 可选项

点击CodeWarrior IDE 的菜单Project 下的make 菜单，就可以对工程进行编译和链接了。整个编译链接过程如图2.9 所示：

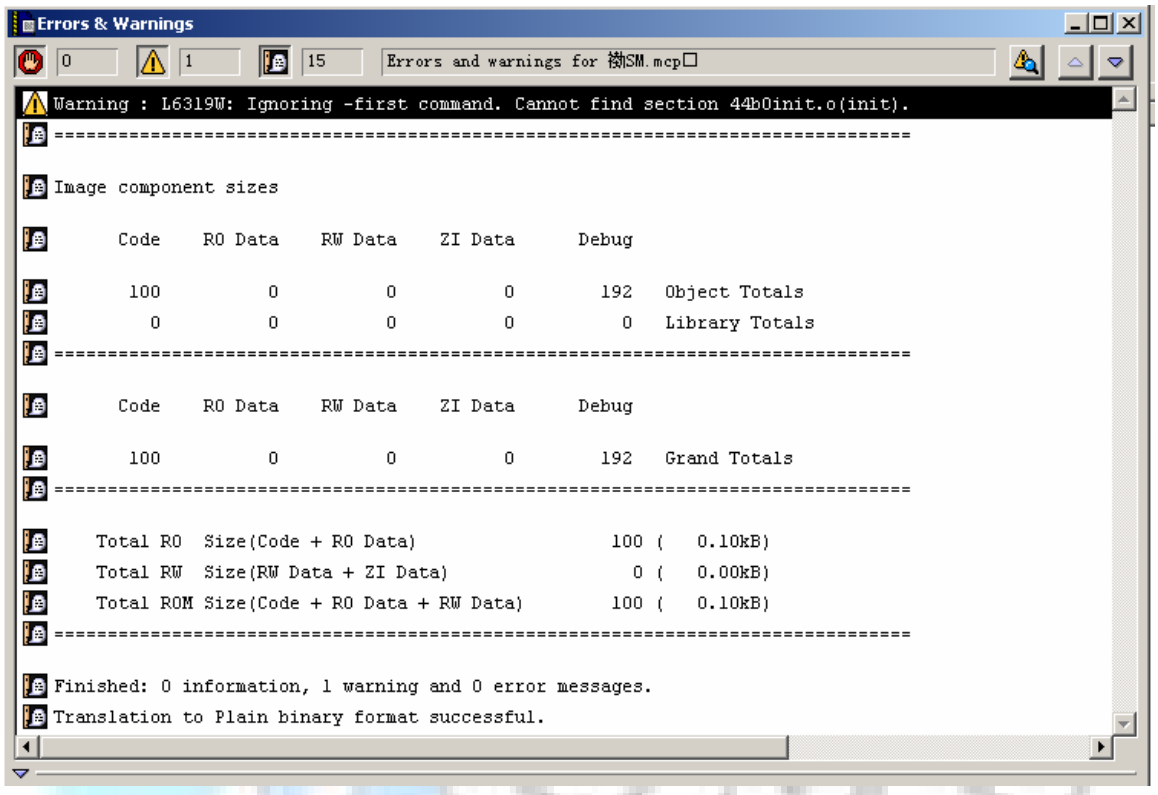


图4.4-10 编译和链接过程

在工程所在的目录下，会生成一个名为：工程名_data 目录，在本例中就是ASM_data目录，在这个目录下不同类别的目标对应不同的目录。在本例中由于我们使用的是DebugRe 目标，所以生成的最终文件都应该在该目录下。进入到DebugRel 目录中去，读者会看到make 后生成的映像文件和二进制文件，映像文件用于调试，二进制文件可以烧写到S3C44B0x 的Flash 中运行。

4.2 用 AXD 进行代码调试

4.2.1 安装并运行 ARM-JTAG 调试代理 ARM7 请看多媒体演示程序

4.2.2 为 ARM-JTAG 调试代理正确配置 AXD DEBUGGER 请看多媒体演示程序

4.2.3 使用 SUPERJTAG 在 ADS1.2 环境下进行仿真调试

关闭并重新启动 AXD Debugger，点击菜单 File/Load Image，找到您想调试的调试目标代码（*.axf 格式），打开它就启动目标代码下载，这时会出现进度条，下载完毕就可以单步或者全速调试了，调试过程中可以看 CPU 各寄存器，也可以设置断点，可以单步，全速等等调试功能。

第五章．学习板相关程序烧写及 BIOS 使用方法

5．1 BIOS 烧写方法

注意：在使用JTAG前请确定你的并口设置为EPP

1、LJD-44B0XDVK-I 开发板出厂之前BIOS 已经写好，如果在您的使用过程中不慎冲掉，按以下步骤重新烧写：

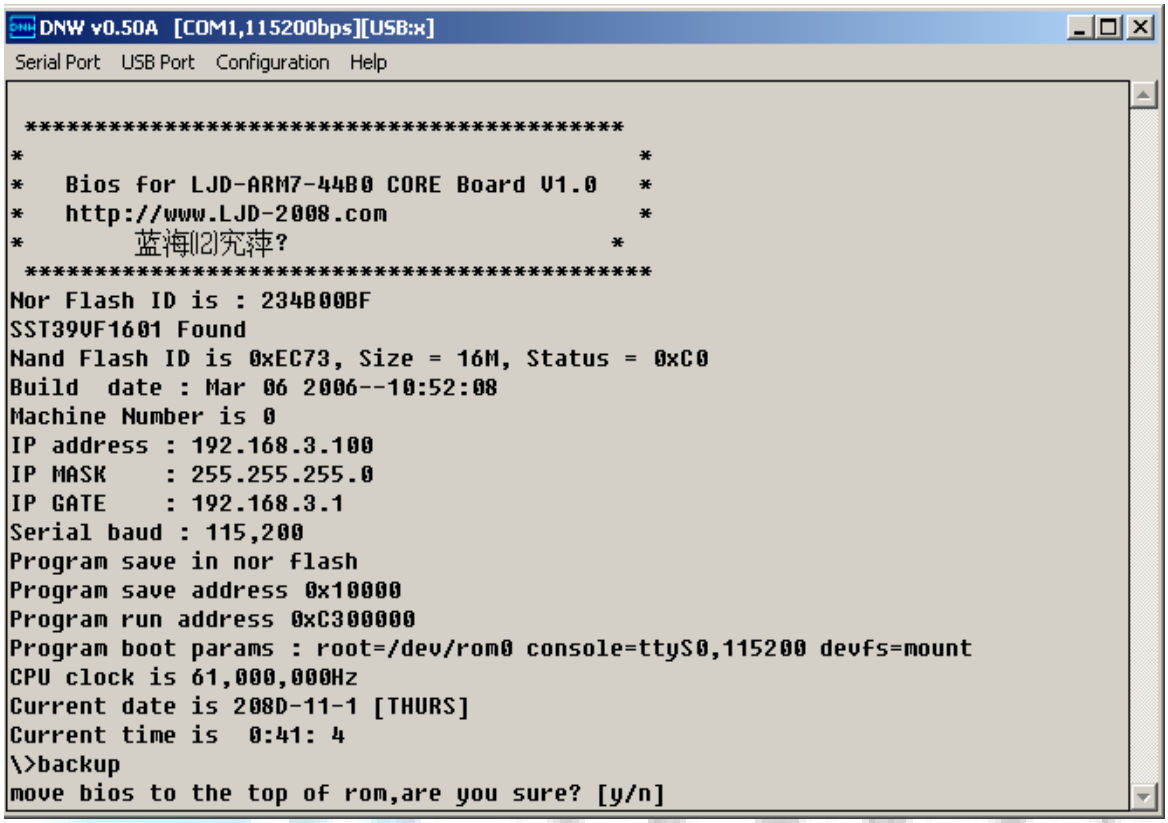
2、将光盘里的“ 目标代码 ”下的LJD-44b0XDVK bios子目录copy 到硬盘上，譬如目录c：\ 。

3、连接SUPERJTAG，将20 芯排线连接开发板和SUPERJTAG 板的Wiggler 接口JP2 上面，打开电源；在FLASHPGM 里点击program ;在Browse 中找到“ LJD-44b0XDVK bios ”文件夹下的LJD-44B0XDVK-I_BIOS.hex 文件；点击Erase Target Flash Sector(s.) Before Pro 前面的小窗口，使其前面打勾；点击Program 按键开始烧写，5 秒钟左右即可烧完；然后点击Close 按键关闭烧写界面，这时LJD-44B0XDVK-I 开发板上的蜂鸣器会发出“ 嘀 ”一声响

（如果烧完bios 后蜂鸣器没有发出响声，请检查JUP1是否插有短路器，必须插短路块才能响。）

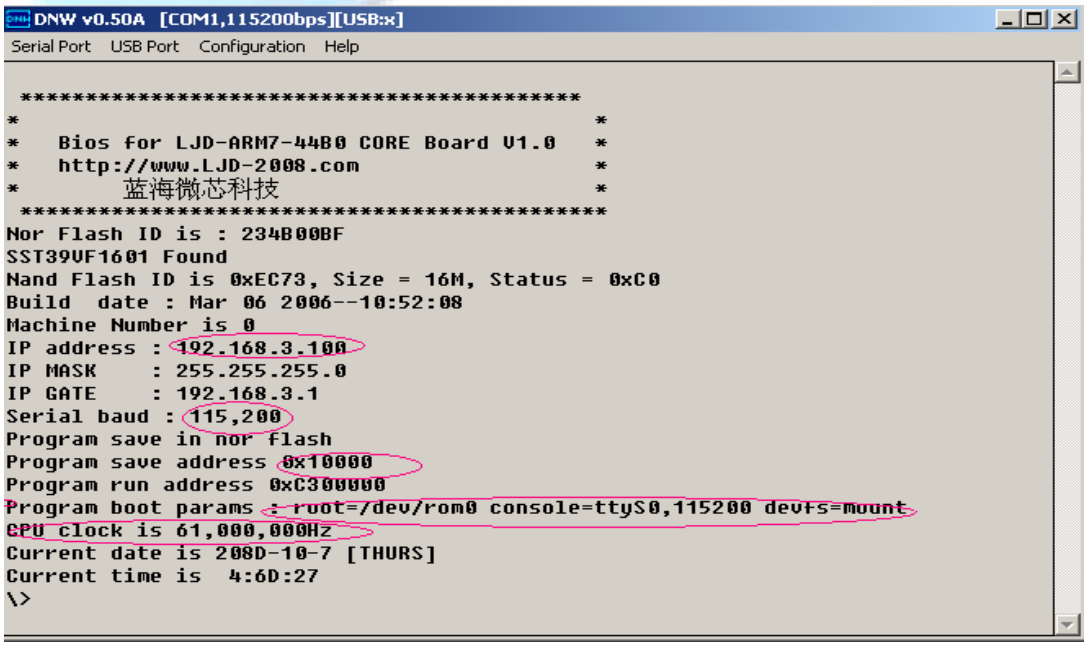
5．2 BIOS 备份

1、打开DNW,连接好串口connect；
2、在Configuration 下点击options 确定超级终端115200；
3、重新上电或者按复位键，超级终端会出现提示信息；
4、在提示符下输入backup，回车，
5、在提问下输入y，屏幕会出现BIOS 拷贝并编程成功的信息；



6、原来在flash 0 地址的Bios 就会拷贝到flash 里以1f0000 地址开始的空间里。

7、进行默认系统设置，在BIOS 命令提示窗口下敲入命令defset 然后回车，按复位键后可以看到如下的提示信息，其中圈了红色圆圈的部分使默认系统设置：



5．3 用 BIOS 通过网口烧写 UC/OS-II 应用程序到 FLASH

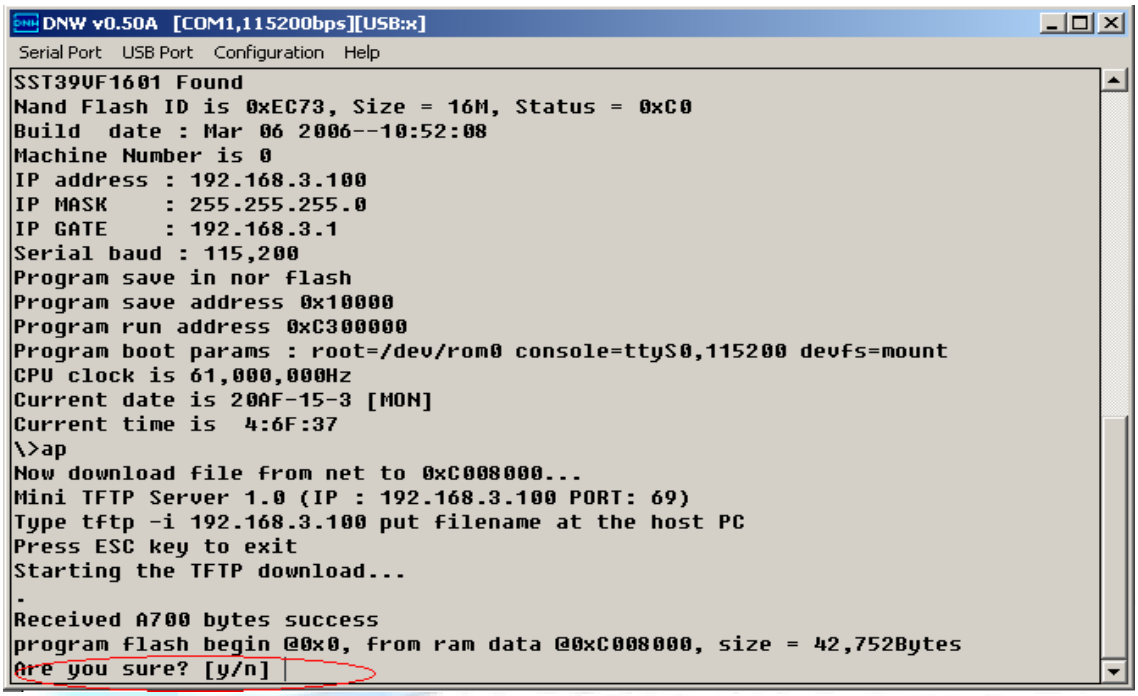
1、LJD-44B0XDVK-I_BIOS 烧写完成后，LJD-44B0XDVK-I开发板重新上电LJD-44B0XDVK-I_BIOS 就接管

了系统， 它初始化了CPU 的串口、网口、内存等。可以在超级终端输入一些指令来完成一些工作。

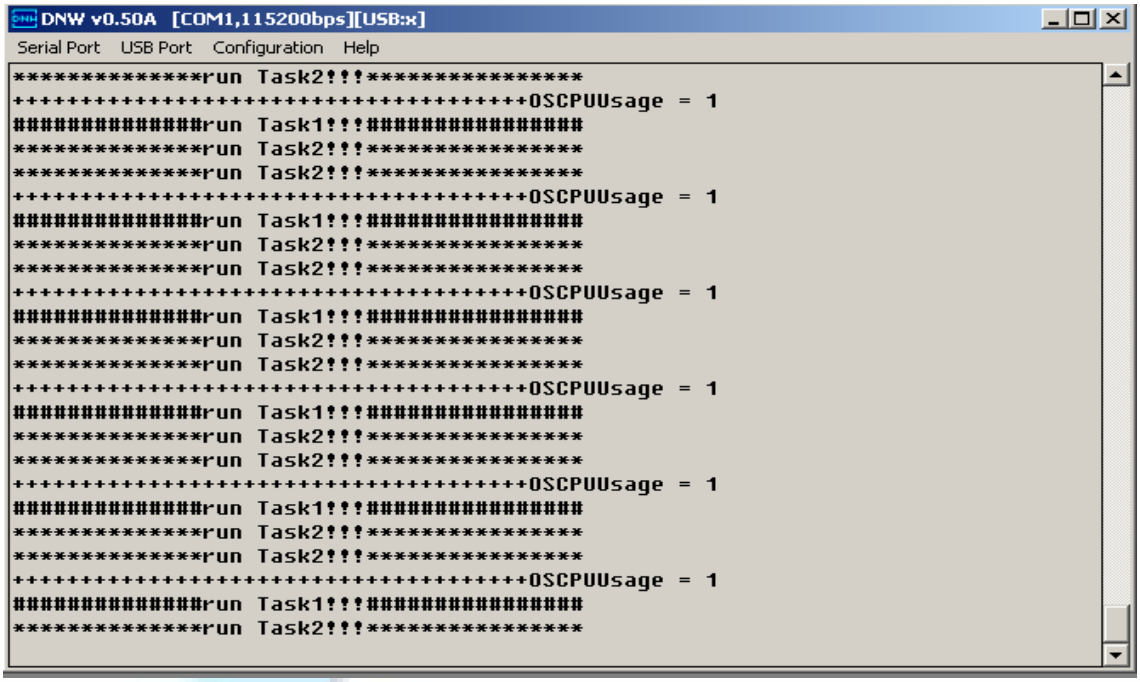
2、先用BIOS 烧写一个UC/OS-II 程序代码。连接好网线和串口线，打开电源；

3、在DNW.exe 里面的BIOS 命令提示符下输入命令ap 并回车；

4、在“目标代码”下，双击批处理文件uCOS-II.BAT，可以看到通过网口下载了0xA700个字节的目标代码文件到板子上面了，然后在dnw.exe 里输入y 并回车，就将uC/OS-II 程序烧写到0 地址（但没有修改0 地址的跳转指令），提示信息如下：



5、现在可以关闭电源重新上电或者是直接按复位按键，当超级终端出现提示符/>时，输入命令boot 并回车，当系统提示boot from flash, are you sure ?[y/n]时键入“ y ” 即启动刚烧入的UC/OS-II 程序。这时我们可以在LJD-44B0XDVK-I 板子在DNW显示如下画面。即任务TASK1到TASK3轮流执行，同时输出CPU使用率。这时，BIOS 已将控制权交给了应用程序，超级终端受应用程序控制。相关提示信息



6、此时要进入BIOS 的命令提示符下，只能按复位键重新启动BIOS。

5．4 用 BIOS 通过网口快速运行和调试 UC/OS-II 应用程序

有经验的嵌入式开发工程师会发现，其实很少时候需要用到仿真器来单步调试，大多数情况下都是全速运行代码，但BIOS 里有命令可以支持这种快速调试。在BIOS 的命令提示符下执行命令netrun（更简洁快速的命令是g），然后点击批处理文件
uCOS-II.BAT，可以看到目标代码文件uCOS-II.bin 很快就通过网口下载到
SDRAM 中从0xc008000 开始的地址，然后马上从这个执行该程序。

由于网口的数据传送速度是10M，而MULTI－ICE 仿真器使用的并口数据传送速度只有几百K，所以使用上面的方法，下载并全速运行程序的速度比MULTI－ICE 仿真器快很多。正因为如此，所以很少使用MULTI－ICE 仿真器，仅只在极少数需要单步跟踪调试代码的时候才用到它，绝大多数情况下都是在BIOS 里面敲一个g，然后点击一个批处理就搞定调试过程。

5．5 BIOS 相关命详解

LJD-44B0XDVK-I_BIOS 具有启动、引导，下载、烧写，设置日期、时间，设置工作频率等多种功能，并且支持各种参数的存储和自动调用。

可以用FLASHPGM 等软件将BIOS 烧写到FLASH 中去，BIOS 的自身驻留地址位于NOR FLASH的0x1f0000处，系统参数保存在0x1ff000 以上区域中。所以在烧写完BIOS，上电复位后先要执一定要执行backup 命令把BIOS 本身拷贝到NOR FLASH 的高端1f0000 去。

下面BIOS 命令的汇总列表：

BIOS 命令	命令行提示信息	功能说明
help	show this list	显示BIOS 命令列表
?	help	显示 BIOS 命令列表
date	show or set current date	显示或设置开发板时钟的当前日期

time	show or set current time	显示或设置开发板时钟的当前时间
setweek	set weekday	设置星期信息
clock	show system running clock	显示系统当前运行速度
setmclk	set system running clock	设置系统当前运行时钟
chguart	change uart(0/1)	改变串口（0/1）
setbaud	set baud rate	设置串口通信波特率
ipcfg	show or set current IP address	显示或设置当前开发板 IP 地址
netload	download file by net	通过网络下载文件
netrun	download file by net and run	通过网络下载文件并运行该文件
g	download file by net and run	通过网络下载文件并运行该文件
comload	download file by uart	通过串行口下载文件
comrun	download file by uart and run	通过串行口下载文件并运行该文件
rx	download file by xmodem	通过 XMODEM 下载文件
rxrun	download file by xmodem and run	通过 XMODEM 下载文件并运行
prog	program flash	对 FLASH 存储器编程
ap	download file and program it to flash	下载文件并编程写入 FLASH 存储器
backup	move bios to the top of flash	拷贝 BIOS 代码到 FLASH 存储器高端
copy	copy flash from src to dst address	从源地址到目标地址拷贝 FLASH 内容
boot	boot from flash	从 FLASH 存储器重启动
run	run program	运行程序
move	move data from addr1 to addr2	从 ADDR1 转移数据到 ADDR2
mrun	move data form prog_s_addr to prog_r_addr and run	从源地址到目的地址移动数据并运行
md	show memory data	显示内存数据
memd	show 8/16/32bits memory	按照字节/半字/字格式显示内存数据
mems	set 8/16/32bits memory	按照字节/半字/字格式设置内存数据
senv	save enviroment value to flash	在 FLASH 存储器保存环境变量
machine	set machine number	设置机器序列号
setpa	set program save(run) address	设置程序存储及运行地址
setbp	set program boot parameters	设置程序启动参数
bootkey	set key to autoboot	设置自动重启的按键
nfpart	set nand flash partitions	设置 NAND FLASH 分区
nferase	erase nand flash partition	清除 NAND FLASH 分区
nfprog	program nand flash	NAND FLASH 编程
nfload	load program from nand flash	从 NAND FLASH 载入程序
defset	default setting for ucLinux	恢复到运行 uCLinux 的默认设置

（注：以下命令所带参数中地址和长度都属 16 进制，不必在前面加 0x）

help 和?可以列出所有命令并给出简单的说明；

date 命令可以显示和设置当前日期，只输入 date 命令则显示日期，输入 date 2004-6-8 则设置当前日期为 2004 年 6 月 8 日。

time 命令可以显示和设置当前时间，只输入 time 命令则显示时间，输入 date 14：4：30 则设置当前时间为 14：4：30。

setweek n 可设置星期几，n 从 1 到 7 表示星期一到星期日。

clock 可以显示当前的工作频率。

setmclk 可以改变 CPU 工作频率，具体参数设置可见芯片手册，注意不要使频率超出工作范围。频率参数属于可以保存和调入的参数，这次设置和保存后下次复位 BIOS 会自动调入这写参数初始化 CPU。

setbaud 可改控制串口的波特率，改完后要在 PC 上相应改变串口通讯波特率后再敲回车。

ipcfg 可显示和修改 tftp 下载时所用的 IP 地址，只输入 ipcfg 则显示当前 IP 地址，输入 ipcfg 192.168.2.223 则将 ip 地址改为 192.168.2.223。

netload 启动 tftp 接收，若没带地址参数，则使用缺省下载地址 0x0c008000,若指定地址，下载数据保存到指定地址开始的 SDRAM 中去，如 netload c300000。启动 tftp 接收后，要在 PC 端执行 tftp 下载程序，在 win2000 或 winxp 下，直接输入 tftp -i xxx.xxx.xxx.xxx put 文件名即可；在 win98 下，使用 CDROM 里所带的 tftp 程序；在 linux 下，使用 CDROM 里所带的 tftpcmd 程序。注意进行 tftp 传输时要保证 PC 机和开发板处于同一个 IP 段内。

netrun 或者是 g 启动 tftp 接收完数据后会自动运行下载到的程序，缺省下载地址和指定参数同 netload。

comload 启动串口下载（DNW 程序的串口下载），缺省下载地址和指定参数同 netload.

comrun 启动串口下载（DNW 程序的串口下载）并在接收完数据后自动运行下载的程序，缺省下载地址和指定参数同 netload。

rx 启动 XMODEM 方式下载，可在超级终端内选择 1K XMODEM 或 XMODEM 发送数据到开发板上，缺省下载地址和指定参数同 netload。

rxrun 在启动 XMODEM 方式接收完数据后自动运行下载到的程序，缺省下载地址和指定参数同 netload。

prog 可以烧写 NOR FLASH，目前支持 SST39VF160。prog 命令完整的参数是 prog addr1 addr2 length [-no0],其中 addr1 是要烧写的 FLASH 的地址，大于等于 0，小于 200000，字对齐，addr2 是 sdram 中要烧进 flash 的数据区起始地址，length 是要烧写的长度，-no0 表示要把数据烧进 Nor Flash 0 地址开始的地方时，是否修改 0 地址的指令，因为 CPU 复位总是从 0 开始执行的，当用 Nor Flash 启动时，若用 prog 命令将下载到的程序烧入 Nor Flash 0 地址开始的地方并在命令最后指定-no0,那么在复位后，就不会再运行 Bios 而直接启动用户程序了，若不在 prog 命令最后加-no0，则 BIOS 可以烧写 Nor Flash 0 地址的数据前，将 0 地址的指令改为直接跳转到 0x1f0000 处即 Bios 的驻留地址，并保存原程序 0 地址将要跳转到的地址,以后在执行 boot 指令时再跳转过去执行用户烧入的程序。运行 Bios。在运行 Bios 下载完数据后，也可不带参数直接执行 prog 命令，缺省的 Nor Flash 地址是用户程序存储地址 prog_s_addr（见后面 setpa 命令），sdram 中数据起始地址和数据长度在接收成功后自动设定了。

ap 指令自动下载完数据并将数据烧写到 nor flash 的 0 地址处，缺省为 tftp 下载，指定-c 表示串口下载（DNW 方式），-x 表示 XMODEM 下载，-b 表示不修改 0 地址的指令。

backup 可用在第一次烧写完 BIOS 到 Nor Flash 0 地址后上电执行时将 Bios 本身拷贝到 0x1f0000 处。

copy 将 Nor Flash 某地址的数据拷贝到另一地址。

boot 可运行用户通过 BIOS 下载烧写到 0 地址并修改过 0 地址跳转地址的程序，见 prog.

run 可运行存储器中的程序，缺省地址就是缺省下载地址，也可指定运行地址。

move addr1 addr2 size 可将存储器中 addr1 开始的长度为 size 的数据拷贝到 addr2 开始的地址去。

mrun 可自动执行 move 的过程并运行程序，比如在 FS44B0 中我们将 uClinux 内核保存在 Nor Flash 的 0x10000 开始的地方，长度为 800K，它的运行地址是 0x0c300000,那么 mrun 就可以完成拷贝的操作

并直接运行。mrun 内部使用的参数见 setpa 命令。

md 显示存储器中的数据，可以带地址参数。

memd 可显示单个存储器单元中的内容，-c 参数表示 8 位数据，-s 参数表示 16 位数据,-l 参数表示 32 位数据，后面跟存储器地址。

mems 可修改单个存储器单元中的内容，-c,-s,-l 参数同上，后面跟存储器地址和要写入的内容。

machine 可设置机器号，适用于 linux,此参数可保存。

setpa 有几个参数

Usage : setpa -s[-r][-i][-ni][-nor][-nand] [address]

```
-s    save address
-r    run address
-i    initrd save address
-ni   disable initrd
-nor  use nor flash to save
-nand use nand flash to save
```

其中—s 表示用户程序在 FLASH 中的存储地址，如上面所说的将 uClinux 内核保存到 Nor Flash 的 0x10000 处，为使 mrun 正确运行，我们就要设置 setpa -s 10000

—r 表示用户程序的运行地址，如上面所说的将 uClinux 内核的运行地址是 0x0c300000,为使 mrun 正确运行，我们就要设置 setpa -r c300000

—i 表示使用 initrd(对于 linux 或 uClinux),它的存储地址是多少。

—noi 表示取消 initrd。

—nor 表示用户程序存储在 Nor Flash 中，-nand 表示用户程序存储在 Nand Flash 中。注意使用 Nand Flash 存储时，前述保存地址 1000 表示 Nand 分区 1，2000 表示 Nand 分区 2，依此类推，Nand 分区见 nfpart 命令。

setpa 设置的参数都是可以保存的。

setbp 可以设置启动命令（对于 uClinux 和 linux），可以保存。

Usage : setpa -s[-r][-i][-ni][-nor][-nand] [address]

```
-s    save address
-r    run address
-i    initrd save address
-ni   disable initrd
-nor  use nor flash to save
-nand use nand flash to save
```

—s 表示 mrun 运行的程序是存储器在 flash 的什么位置，
对于 nor flash 是 nor flash 中的地址，
对于 nand flash 1000 表示分区 0,2000 表示分区 2.

—r 表示存储的程序要读到 sdram 中什么位置再运行.对于 uClinux 是 c300000.

—i 表示 initrd 存储在 flash 的什么位置,如同-s.

—ni 表示取消 initrd

—nand 表示用 NAND FLASH 作为内核及 initrd 的存储介质.

bootkey 可设置 BIOS 复位运行后检查哪个按键状态来自动启动存储在 Flash 中的用户程序，即自动调用 mrun 指令，按键编号 1~4，状态 0 表示低启动，1 表示高启动。比如要在复为后检测到按键 3 为低时启动，可执行 bootkey 3 0。此参数也可保存，注意实现自动启动的前提是先烧写好 Flash 和用 setpa

命令设置好各个参数，bootkey 命令最后可带-b 参数，表示自动运行 boot 指令，缺省情况下是运行 mrun 指令。

nfpart 可在 Bios 中对 Nand Flash 简单分区，比如 Nand Flash 大小是 32M，要分为 0~0x30000,0x30000~0x200000,0x200000~0x800000,0x800000~0x1000000,0x1000000~0x2000000 这样 5 个分区，可以执行 nfpart 30000 200000 800000 1000000 2000000，分区最多为 8 个，分区参数可以保存。

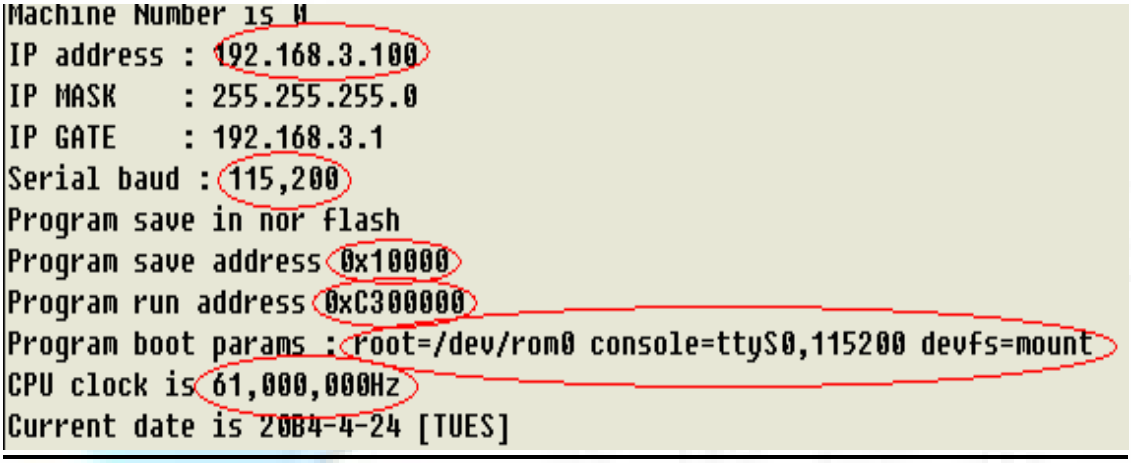
nferase 可以擦除 Nand Flash 分区，块有错误时会有提示。

nfprog 可以将下载的数据写入 Nand Flash 分区，也可指定烧入数据的起始地址和长度，烧写有错误也会有提示。

nfload 可以将 Nand Flash 分区的数据全部读入 sdram 中，可以指定 sdram 地址和 Nand Flash 分区。

senv 命令可以保存所有保存的参数到 Flash 中，下次复位运行 BIOS 后会调入这些参数。

defset 命令可以将运行 uCLinux 所需要的各种默认设置一次性设置好，并且保存起来。默认的设置如下图中红色圆圈所示：



第六章．UClinux 开发环境的建立和使用方法

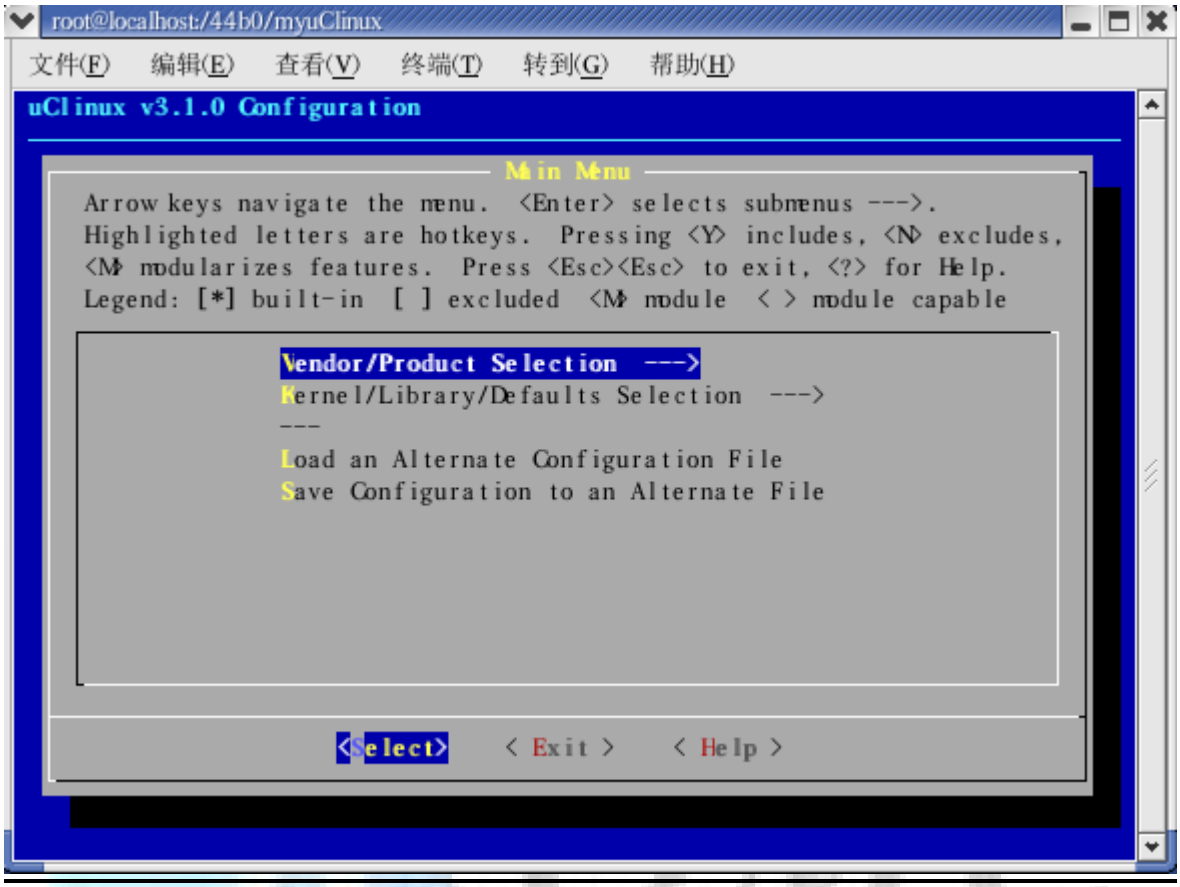
6．1 ucLINUX 开发环境如何建立

获得uClinux-040408-hzh.tar.bz2 的两个途径：

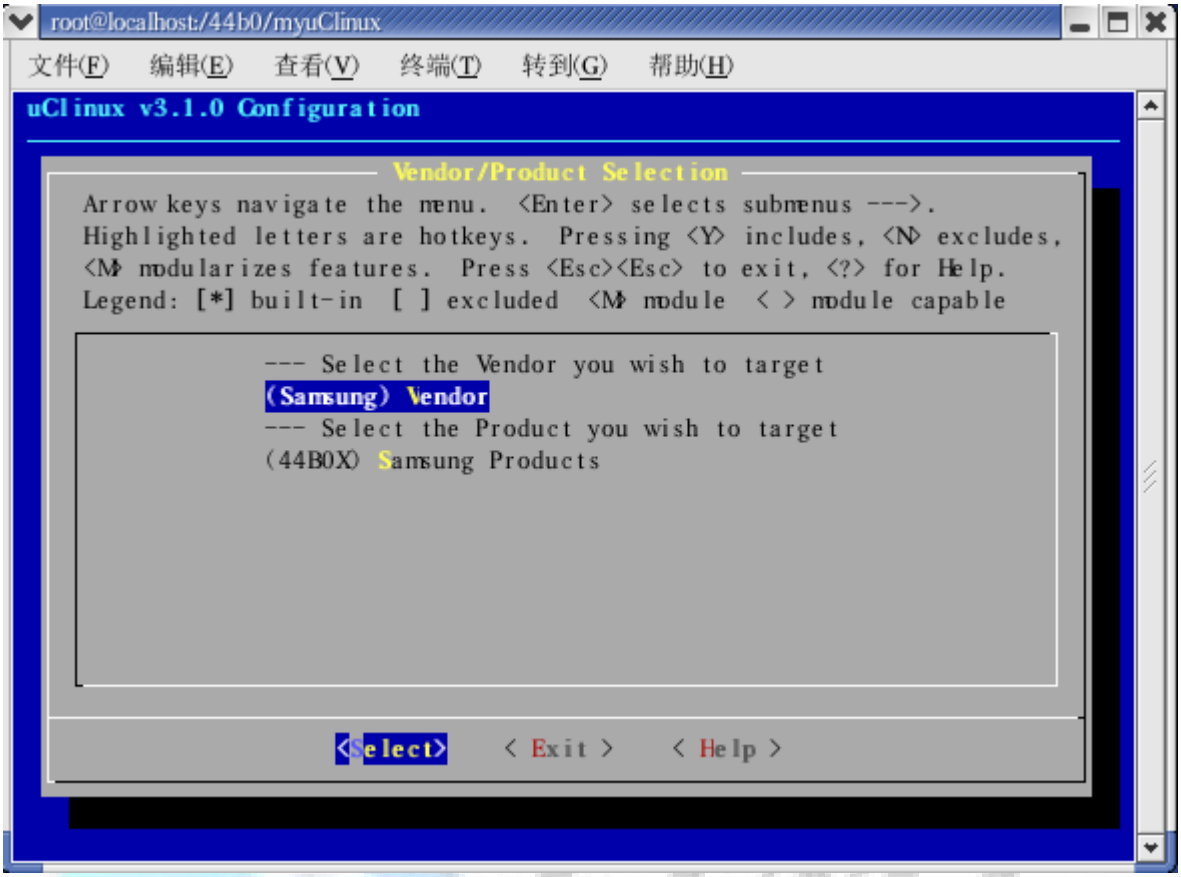
1) 配套光盘uclinux 源码和工具目录下
   —编译工具
   —arm-elf-tools-20030314.sh
2) uClinux 内核源码包
   —LJD44b0X.tar.gz
3) 解压源码包
   —解压uClinux 源码程序包
   —tar xzvf LJD44b0X.tar.gz
   —会生成一个myuClinux的目录
4) 安装交叉编译工具
   —# ls –l arm-elf-tools-20030314.sh
   —该命令显示文件的各种属性，如果该脚本文件属性的不是可执行的，
   —则还需要输入命令：# chmod 755 arm-elf-tools-20030314.sh
   —以将其属性改为可执行属性，然后通过键入命令:
   —#sh ./arm-elf-tools-20030314.sh
   —执行后/usr/local/bin/路径下有gcc, g++, binutils, genromfs, flthdr 和elf2flt 等各种实用工具。

6．2 uCLINUX 内核编译步骤

1) cd myuClinux; 进入myuClinux 目录
2) make menuconfig ; 进入菜单方式配置指令
3) 进入如下界面

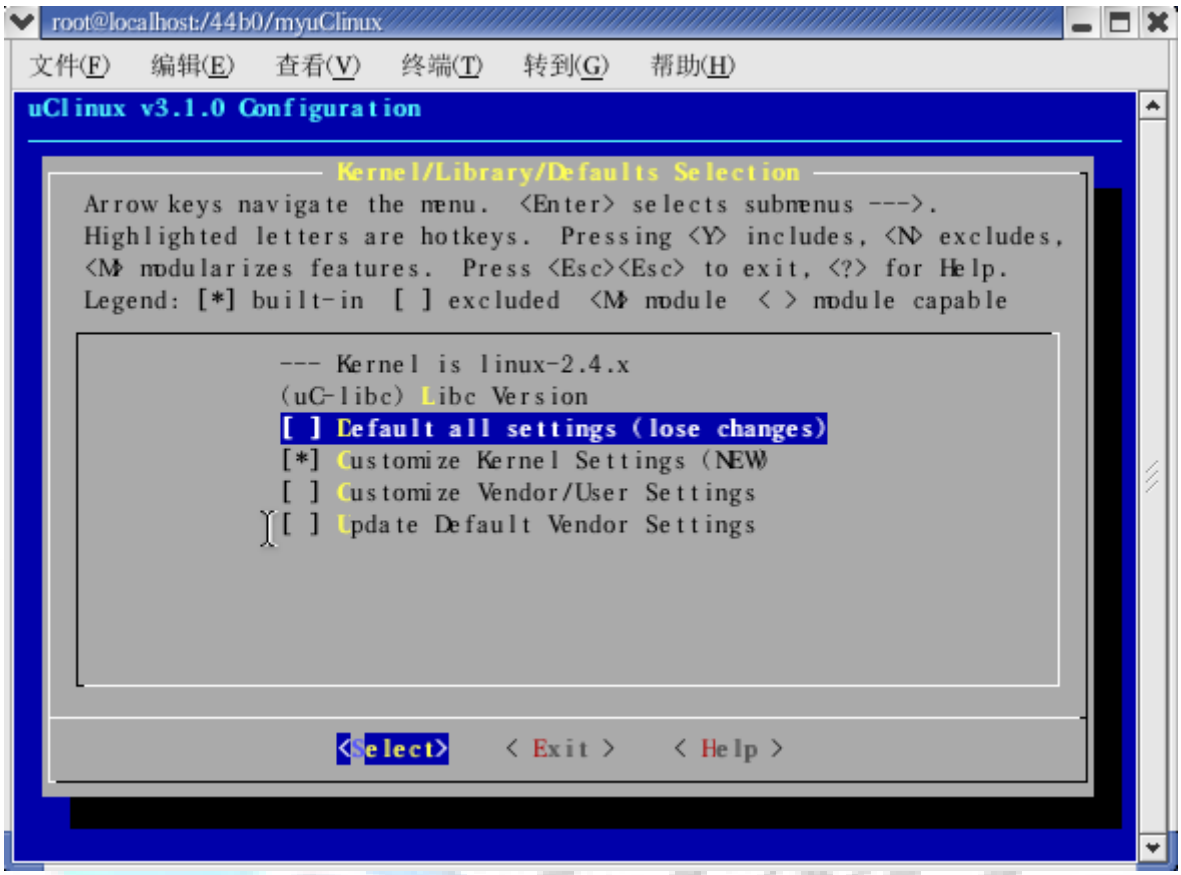


4）选择平台类型和所使用的库,在Vendor/Porduct 下选择S3C44B0X 处理器类型图


root@localhost:/44b0/myuClinux
文件(F) 编辑(E) 查看(V) 终端(T) 转到(G) 帮助(H)
uClinux v3.1.0 Configuration
Vendor/Product Selection
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes,
<M> modularizes features. Press <Esc><Esc> to exit, <?> for Help.
Legend: [*] built-in [ ] excluded <M> module < > module capable
--- Select the Vendor you wish to target
(Samsung) Vendor
--- Select the Product you wish to target
(44B0X) Samsung Products
<Select> < Exit > < Help >

选择CPU类型

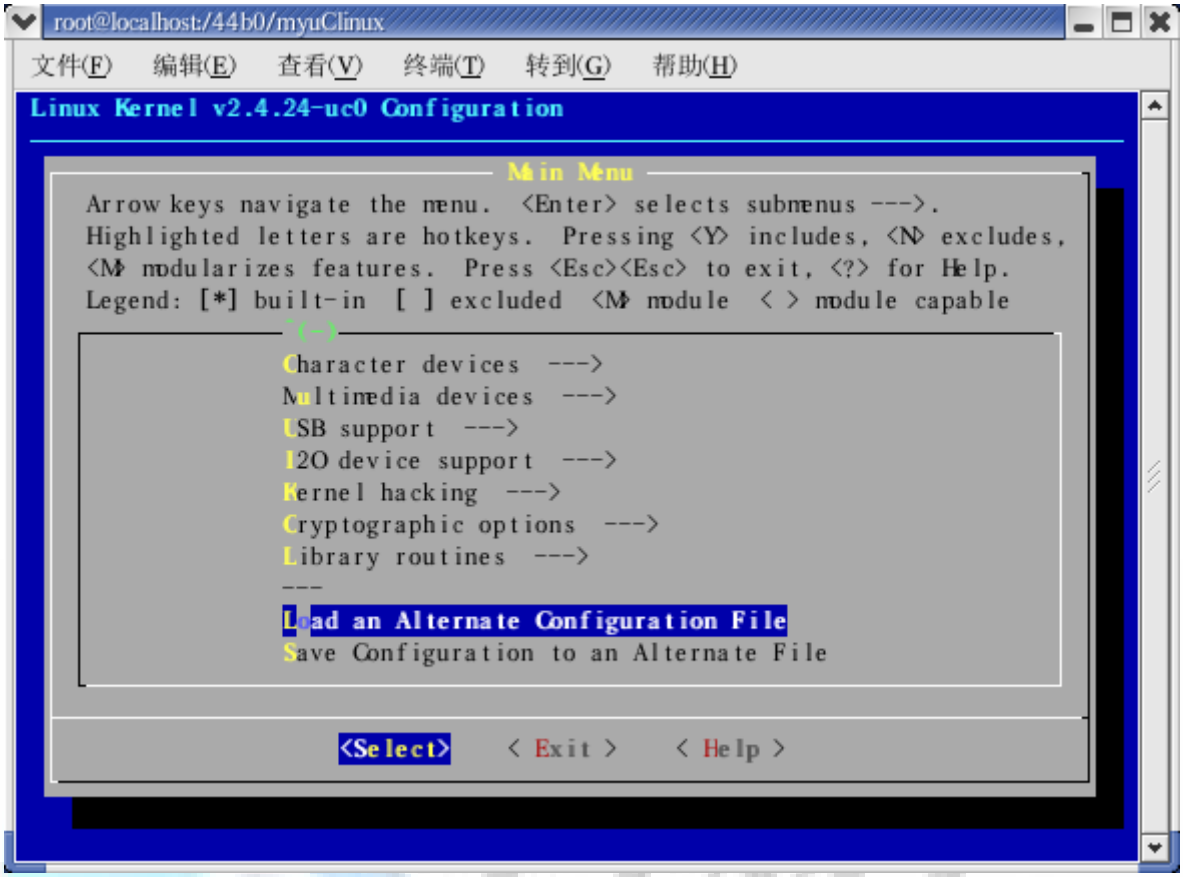
5) 如果想自己配置内核和用户应用，可以选择Customize Kernel Settings (NEW)和Customize Vendor/User Settings(NEW)不选，则按默认的配置


root@localhost:/44b0/myuClinux
文件(F) 编辑(E) 查看(V) 终端(T) 转到(G) 帮助(H)
uClinux v3.1.0 Configuration
Kernel/Library/Defaults Selection
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes,
<M> modularizes features. Press <Esc><Esc> to exit, <?> for Help.
Legend: [*] built-in [ ] excluded <M> module < > module capable
--- Kernel is linux-2.4.x
(uC-libc) Libc Version
[ ] Default all settings (lose changes)
[*] Customize Kernel Settings (NEW)
[ ] Customize Vendor/User Settings
[ ] Update Default Vendor Settings
<Select> < Exit > < Help >

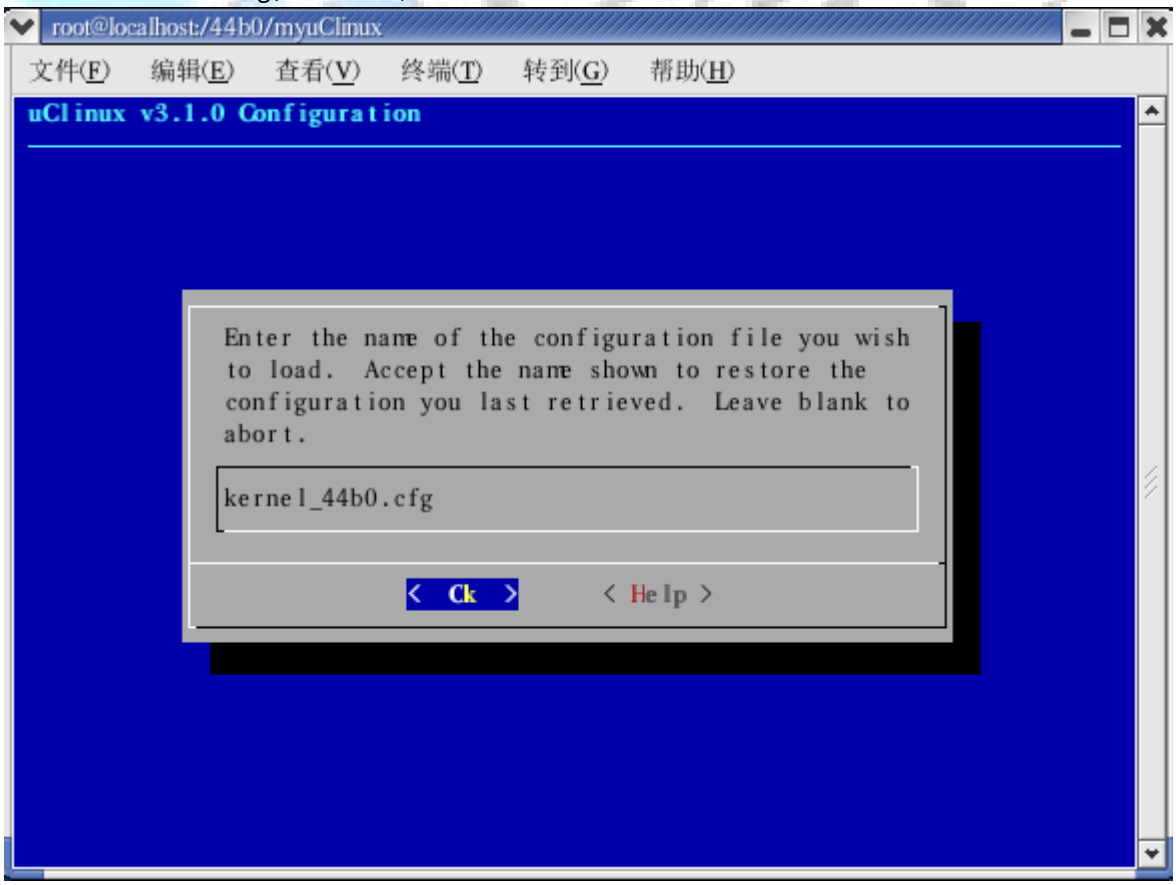
选库文件和自定义内核操作

这里我们选择“customize kernel settings”一项，退出并保存。

6）接下来的配置栏中，选择Load an Alternate Configuration File选项，



接着输入kernel_44b0.cfg,然后确认,退出配置栏



接下来键入命令:make dep 用以建立依赖关系。

7)键入命令：make clean ;清除以前构造内核时生成的所有目标文件

8）键入命令： make lib_only ;编译库文件

9）键入命令： make user_only ;编译用户应用程序文件

10）键入命令：make romfs :生成romfs文件系统

11）键入命令：make zImage ;生成内核映像

12）键入命令：make

通过各个目录的Makefile 文件进行，会在各目录下生成一大堆目标文件。

上述步骤完成后，就完成了对uClinux 源码的编译工作。整个编译过程视计算机运行速度而定。

当内核的编译工作完成之后，会在mylinux/images的目录看到Image.ram ，Image.rom和romfs.img文件。

Image.rom,和romfs.img文件烧写到FLASH存储器中运行，怎样烧写，请看6.3节关于uClinux 的烧写。

6．3 uCLINUX 如何烧写

6.3.1 烧写中断向量表 BOOT.BIN

将光盘里的“ uCLinux_LJD-44B0XDVK-I ”文件夹复制到硬盘上;在超级终端里输入ap(注意ap 是小写)然后在“ uCLinux_LJD-44B0XDVK-I ”文件夹下双击批处理文件boot.bat:（将boot.bin 通过网口下载到SDRAM）；在超级终端里等待下载结束，然后执行命令:输入”y”后按回车键。这部操作是将中断向量表烧写到0地址但不覆盖跳转指令

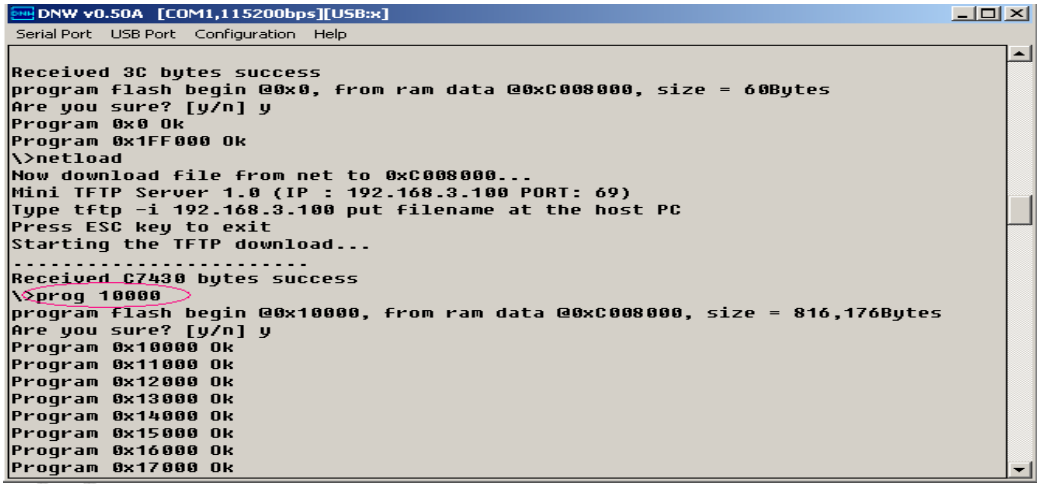
6.3.2 烧写 UCLINUX 内核

在超级终端里输入netload，(注意netload 是小写)

然后在“ uCLinux_LJD-44B0XDVK-I ”文件夹下双击批处理文件LJD-44B0XDVK-I_zImage.bat:(将zImage 通过网口下载到SDRAM)

在超级终端里等待下载结束，然后执行命令:

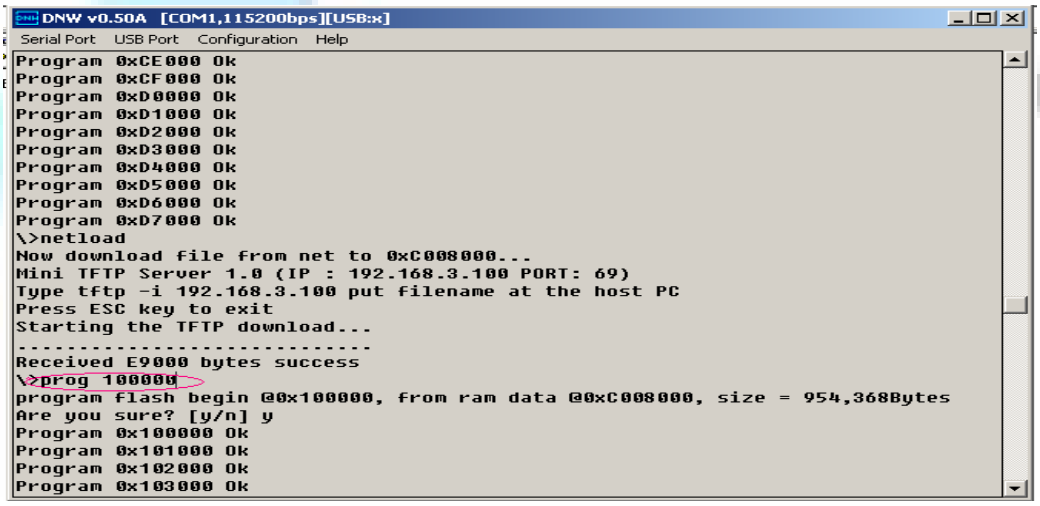
\> prog 10000 回车然后按“ y ”即可把zImage 烧写到FLASH 10000 地址。



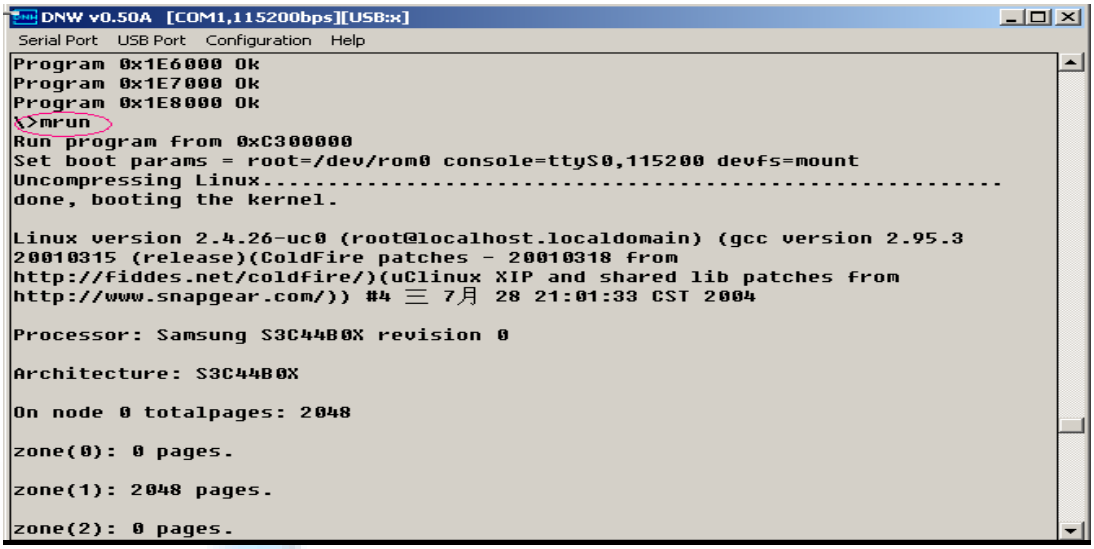
6.3.3 烧写 UCLINUX 根文件系统

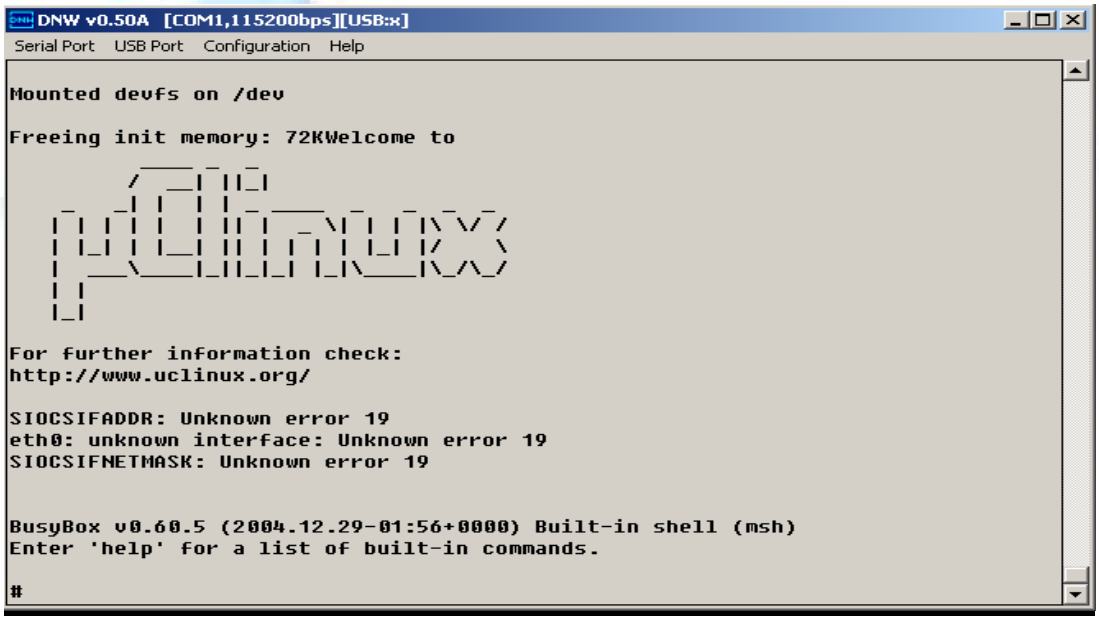
1、在超级终端里输入netload，(注意**netload** 是小写)然后在upload 里双击批处理文件

2、LJD-44B0XDVK-I_romfs_img.bat： (将romfs.img 通过网口下载到SDRAM)，在超级终端里等待下载结束，然后执行命令:\> prog 100000 回车然后按“ y ”即可把此文件烧写到FLASH 100000 地址。



6.3.4 输入命令 MRUN 运行 ucLINUX





6．4 测试目标

启动目标系统的uClinux，通过超级终端，在命令提示符下输入：

ifconfig eth0 192.168.3.100

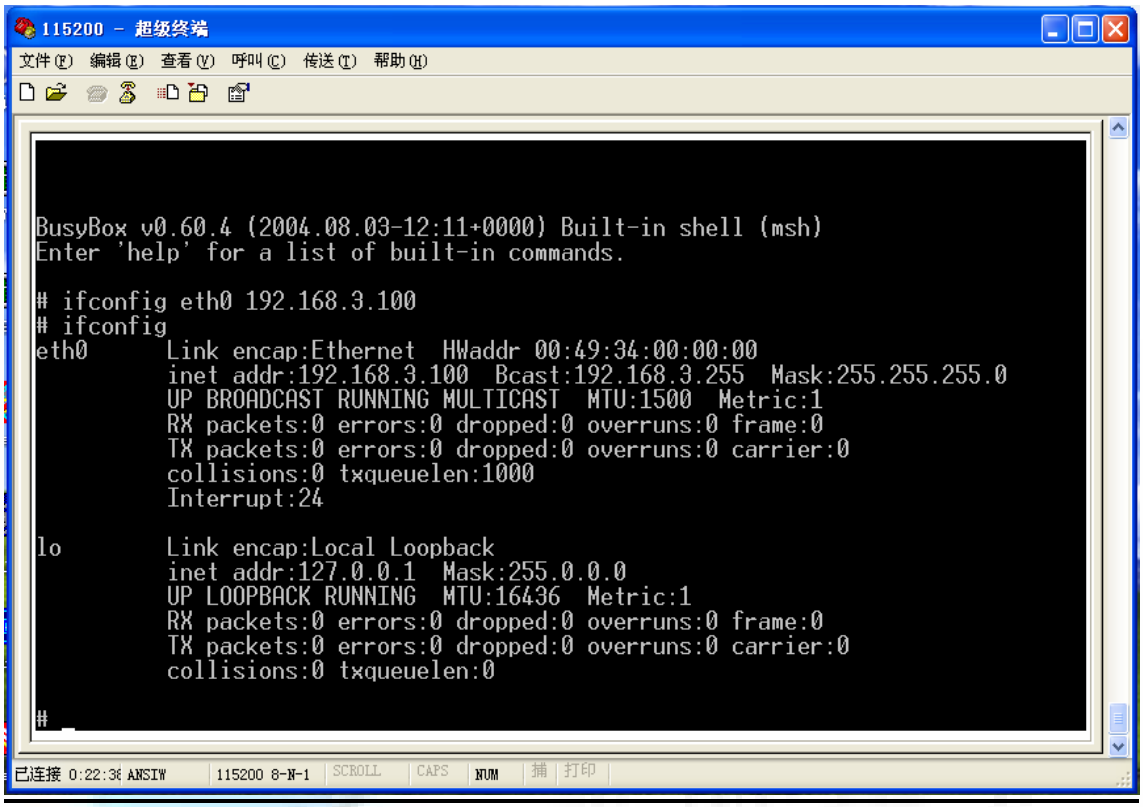
ifconfig 命令用于显示及设置目标系统的网卡配置，例如，IP 地址，子网掩码，IRQ 及IO Port 等。在上述命令中，参数eth0 代表目标系统的网络设备，IP 地址192.168.3.50 为目标系统的IP 地址，注意应与宿主机在同一网段内(此时宿主机的IP 地址为：192.168.3.80),当然目标系统和宿主机的IP 地址可

以自由的设定，只要保证它们在同一个网段就行了

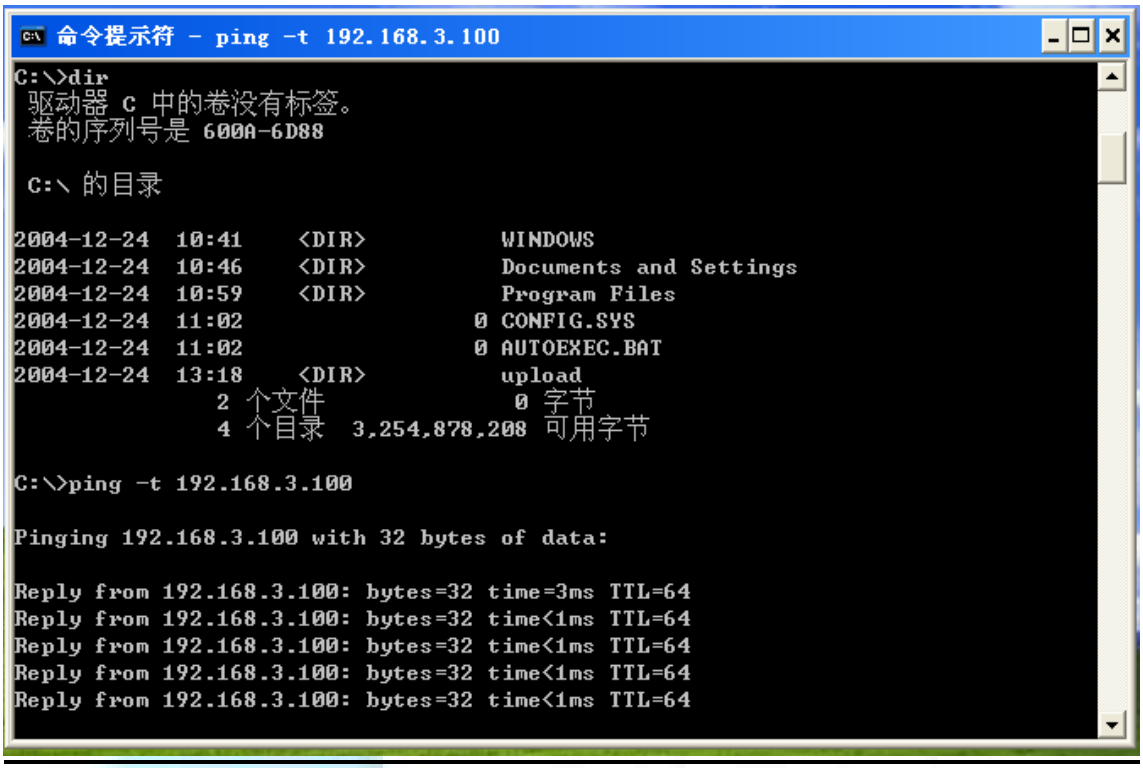
执行如下命令：

ifconfig –all

就可以看到目标系统的IP 地址配置信息了，显示信息见下图：



在PC 机上运行ping 192.168.3.100 ，PING 成功后显示信息如下图：



第七章．学习板常见故障及解决方法

7.1 如何重新烧写 BIOS?

请参看第五章如何烧写 BIOS ”

7.2 为何超级终端不能输入？

将超级终端 com1 或 com2 属性/端口设置中的流量控制设置为---无

7.3 为何超级终端里，BIOS 不接受 AP 等指令？

BIOS 的指令都是小写，只能输入 ap 等指令

7.4 为何网络 PING 不通？

请将 PC 机的 IP 地址设置为 192.168.3.80 ,子网掩码设置为 255.255.255.0, LJD-44b0xDVK-I 的 bios 启动后，在超级终端里输入 ap 并回车。再在 WINDOWS 里运行 PING192.168.3.100 -t ,即可 PING 通。

7.5 为何 SUPERJTAG 连不上目标板？

这个一般是因为连线没有连好，或者计算机 BIOS 里面没有把并口设置成 EPP 模式，或者是设置不正

确。请记住一定要在 ARM7.exe 里面看到检测到 ARM7TDMI 才说明硬件没有问题。发现有些笔记本无法使用 SUPERJTAG，但台式机一般都没有问题，所以最好用台式机来调试。

7.6 FLASHPGM 无法进行 FLASH 烧写怎么办？

首先请确认您使用的 FLASHPGM 已经解密了，正确设置通讯电缆类型、并口端口、OCD 配置文件等。请参阅 FLASHPGM 如何使用。